Membangun sistem perayapan web yang dapat diskalakan untuk data ESG AWS

AWS Bimbingan Preskriptif

AWS Bimbingan Preskriptif: Membangun sistem perayapan web yang dapat diskalakan untuk data ESG AWS

Table of Contents

Membangun sistem perayapan web yang dapat diskalakan untuk data ESG AWS

Vijit Vashishtha dan Mansi Doshi, Amazon Web Services

Januari 2025 (sejarah dokumen)

Faktor lingkungan, sosial, dan tata kelola (ESG) merupakan pertimbangan penting bagi investor ketika mengevaluasi investasi potensial:

- Lingkungan - Berfokus pada dampak perusahaan terhadap alam. Ini mencakup faktor-faktor seperti emisi karbon, manajemen sumber daya, dan efisiensi energi.
- Sosial — Memeriksa bagaimana perusahaan mengelola hubungan dengan karyawan, pemasok, pelanggan, dan komunitas. Ini mencakup aspek-aspek seperti praktik ketenagakerjaan, keragaman, dan keterlibatan masyarakat.
- Tata Kelola — Melihat kepemimpinan perusahaan, kontrol internal, dan hak pemegang saham. Ini termasuk komposisi dewan, kompensasi eksekutif, dan etika bisnis.

Perusahaan dengan praktik ESG yang kuat semakin dipandang sebagai posisi yang lebih baik untuk keberlanjutan dan profitabilitas jangka panjang. Ada permintaan investor yang meningkat untuk informasi ESG. Perusahaan yang dapat menunjukkan kredensi keberlanjutan mereka melalui data ESG yang andal dan berguna memiliki posisi yang lebih baik untuk menarik modal dan tetap kompetitif. Perusahaan mempublikasikan data ESG melalui berbagai sumber, seperti berita, artikel, dan laporan tahunan. Karena informasi ini tersebar, perayap web dapat membantu Anda mengumpulkan data ini secara efisien.

Panduan komprehensif ini menunjukkan cara menggunakan AWS FargateAmazon Elastic Compute Cloud (Amazon AWS BatchEC2), dan Amazon Simple Storage Service (Amazon S3) untuk membangun pipeline pengumpulan data yang kuat, terukur, dan bertanggung jawab. Ini membahas hal berikut:

- Merancang sistem crawling yang dapat diskalakan dengan menggunakan yang berikut ini: Layanan AWS
  - Fargate atau Amazon EC2 untuk menjalankan aplikasi crawler
  - AWS Batch untuk mengatur pekerjaan perayapan skala besar secara efisien

  - Amazon S3 untuk penyimpanan data yang aman dan tahan lama
- Menerapkan praktik terbaik untuk merangkak etis, termasuk:
  - Menghormati robots.txt dan kebijakan situs web
  - Mengelola pembatasan tarif untuk menghindari situs target yang berlebihan
  - Memastikan privasi data dan penggunaan informasi yang dikumpulkan secara bertanggung jawab
- Mengembangkan crawler Python berbasis yang dioptimalkan untuk infrastruktur AWS
- Mengoptimalkan kinerja crawler sambil mempertahankan standar etika

Audiens yang dituju

Panduan ini ditujukan untuk insinyur data dan arsitek cloud yang ingin mengumpulkan data up-to-date ESG dalam jumlah besar secara efisien dari situs web publik. Hal ini sangat relevan untuk proyek-proyek yang melibatkan analisis pasar, penilaian keuangan berkelanjutan, atau penelitian keuangan.

Hasil bisnis yang ditargetkan

Berikut ini adalah alasan umum perusahaan menggunakan data ESG:

- Manajemen risiko — Data ESG membantu Anda mengidentifikasi dan mengurangi potensi risiko yang terkait dengan masalah lingkungan, sosial, dan tata kelola.
- Daya tarik investor — Banyak investor sekarang mempertimbangkan faktor ESG ketika membuat keputusan investasi. Mereka memandang praktik ESG yang kuat sebagai indikator keberlanjutan dan profitabilitas jangka panjang.
- Manajemen reputasi — Kinerja ESG yang baik dapat meningkatkan reputasi perusahaan di antara pelanggan, karyawan, dan masyarakat umum.
- Kepatuhan terhadap peraturan — Seiring meningkatnya peraturan terkait ESG, mengadopsi praktik ESG membantu perusahaan tetap berada di depan persyaratan kepatuhan.
- Inovasi dan efisiensi — Berfokus pada faktor ESG dapat mendorong inovasi dalam produk, layanan, dan operasi. Ini mengarah pada peningkatan efisiensi dan penghematan biaya.
- Keunggulan kompetitif — Kinerja ESG yang kuat dapat membedakan perusahaan dari pesaingnya dan membuka peluang pasar baru.

- Keterlibatan pemangku kepentingan — Praktik ESG membantu perusahaan terlibat dengan lebih baik dan memenuhi harapan berbagai pemangku kepentingan, termasuk karyawan, pelanggan, dan komunitas lokal.

Arsitektur untuk sistem perayapan web yang dapat diskalakan AWS

Diagram arsitektur berikut menunjukkan sistem perayap web yang dirancang untuk mengekstrak data lingkungan, sosial, dan tata kelola (ESG) secara etis dari situs web. Anda menggunakan crawler Python berbasis yang dioptimalkan untuk AWS infrastruktur. Anda gunakan AWS Batch untuk mengatur pekerjaan crawling skala besar dan menggunakan Amazon Simple Storage Service (Amazon S3) untuk penyimpanan. Aplikasi hilir dapat menelan dan menyimpan data dari bucket Amazon S3.

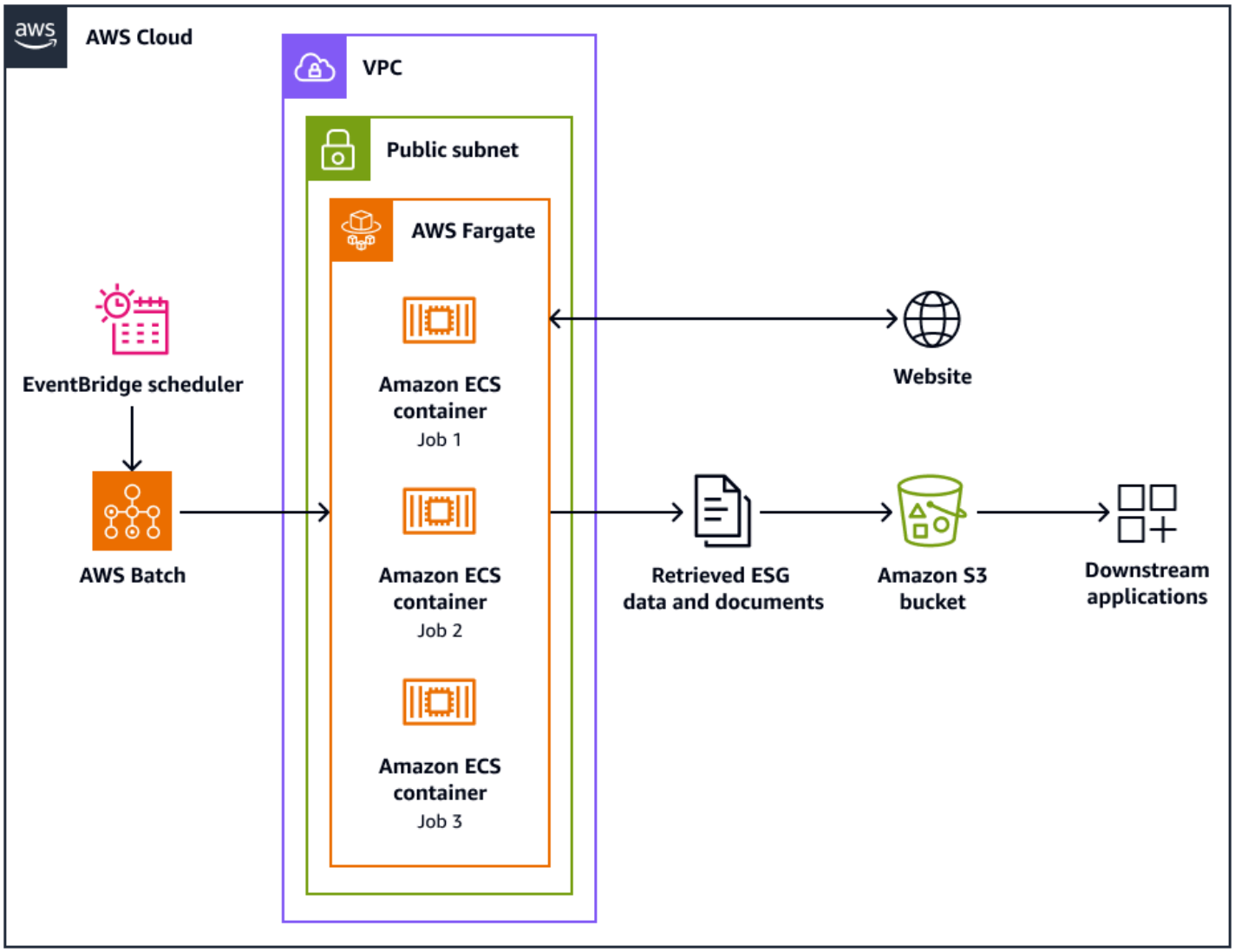

Diagram menunjukkan alur kerja berikut:

1. Amazon EventBridge Scheduler memulai proses crawling pada interval yang Anda jadwalkan.
2. AWS Batch mengelola eksekusi pekerjaan crawler web. Antrian AWS Batch pekerjaan menahan dan mengatur pekerjaan crawling yang tertunda.
3. Pekerjaan perayapan web berjalan di wadah Amazon Elastic Container Service (Amazon ECS) aktif. AWS Fargate Pekerjaan berjalan di subnet publik dari virtual private cloud (VPC).
4. Perayap web merayapi situs web target dan mengambil data dan dokumen ESG, seperti PDF, CSV, atau file dokumen lainnya.
5. Perayap web menyimpan data yang diambil dan file mentah dalam bucket Amazon S3.
6. Sistem atau aplikasi lain menyerap atau memproses data dan file yang disimpan di bucket Amazon S3.

Desain dan operasi perayap web

Beberapa situs web dirancang khusus untuk dijalankan di desktop atau di perangkat seluler. Web crawler dirancang untuk mendukung penggunaan agen pengguna desktop atau agen pengguna seluler. Agen ini membantu Anda berhasil membuat permintaan ke situs web target.

Setelah perayap web diinisialisasi, ia melakukan operasi berikut:

1. Web crawler memanggil `setup()` metode. Metode ini mengambil dan mem-parsing file robots.txt.

> **Note**
>
> Anda juga dapat mengonfigurasi crawler web untuk mengambil dan mengurai peta situs.

2. Web crawler memproses file robots.txt. Jika penundaan crawl ditentukan dalam file robots.txt, crawler web mengekstrak penundaan crawl untuk agen pengguna desktop. Jika penundaan crawl tidak ditentukan dalam file robots.txt, maka perayap web menggunakan penundaan acak.
3. Web crawler memanggil `crawl()` metode, yang memulai proses crawling. Jika tidak URLs ada dalam antrian, itu menambahkan URL awal.

> **Note**
>
> Crawler berlanjut hingga mencapai jumlah halaman maksimum atau kehabisan URLs crawl.

4. Crawler memproses. URLs Untuk setiap URL dalam antrian, crawler memeriksa apakah URL telah dirayapi.
5. Jika URL belum di-crawl, crawler memanggil `crawl_url()` metode sebagai berikut:
   a. Crawler memeriksa file robots.txt untuk menentukan apakah ia dapat menggunakan agen pengguna desktop untuk merayapi URL.
   b. Jika diizinkan, crawler mencoba merayapi URL dengan menggunakan agen pengguna desktop.
   c. Jika tidak diizinkan atau jika agen pengguna desktop gagal merayapi, crawler akan memeriksa file robots.txt untuk menentukan apakah ia dapat menggunakan agen pengguna seluler untuk merayapi URL.
   d. Jika diizinkan, crawler mencoba merayapi URL dengan menggunakan agen pengguna seluler.
6. Crawler memanggil `attempt_crawl()` metode, yang mengambil dan memproses konten. Crawler mengirimkan permintaan GET ke URL dengan header yang sesuai. Jika permintaan gagal, crawler menggunakan logika coba lagi.
7. Jika file dalam format HTML, crawler memanggil `extract_esg_data()` metode. Ini digunakan Beautiful Soupuntuk mengurai konten HTML. Ini mengekstrak data lingkungan, sosial, dan tata kelola (ESG) dengan menggunakan pencocokan kata kunci.

   Jika file tersebut adalah PDF, crawler memanggil `save_pdf()` metode tersebut. Crawler mengunduh dan menyimpan file PDF ke bucket Amazon S3.
8. Crawler memanggil `extract_news_links()` metode. Ini menemukan dan menyimpan tautan ke artikel berita, siaran pers, dan posting blog.
9. Crawler memanggil `extract_pdf_links()` metode. Ini mengidentifikasi dan menyimpan tautan ke dokumen PDF.
10. Crawler memanggil `is_relevant_to_sustainable_finance()` metode. Ini memeriksa apakah berita atau artikel terkait dengan keuangan berkelanjutan dengan menggunakan kata kunci yang telah ditentukan.
11. Setelah setiap upaya perayapan, crawler mengimplementasikan penundaan dengan menggunakan metode. `delay()` Jika penundaan ditentukan dalam file robots.txt, ia menggunakan nilai itu. Jika tidak, ia menggunakan penundaan acak antara 1 dan 3 detik.
12. Crawler memanggil `save_esg_data()` metode untuk menyimpan data ESG ke file CSV. File CSV disimpan di bucket Amazon S3.
13. Crawler memanggil `save_news_links()` metode untuk menyimpan tautan berita ke file CSV, termasuk informasi relevansi. File CSV disimpan di bucket Amazon S3.

14. Crawler memanggil `save_pdf_links()` metode untuk menyimpan tautan PDF ke file CSV. File CSV disimpan di bucket Amazon S3.

Batching dan pengolahan data

Proses crawling diatur dan dilakukan secara terstruktur. AWS Batch memberikan pekerjaan untuk setiap perusahaan sehingga mereka berjalan secara paralel, dalam batch. Setiap batch berfokus pada domain dan subdomain perusahaan tunggal, karena Anda telah mengidentifikasinya dalam kumpulan data Anda. Namun, pekerjaan dalam batch yang sama berjalan secara berurutan sehingga tidak membanjiri situs web dengan terlalu banyak permintaan. Ini membantu aplikasi untuk mengelola beban kerja crawling lebih efisien dan memastikan bahwa semua data yang relevan ditangkap untuk setiap perusahaan.

Dengan mengatur perayapan web ke dalam batch khusus perusahaan, ini berisi data yang dikumpulkan. Ini membantu mencegah data dari satu perusahaan dicampur dengan data dari perusahaan lain.

Batching membantu aplikasi secara efisien mengumpulkan data dari web, sambil mempertahankan struktur yang jelas dan pemisahan informasi berdasarkan perusahaan target dan domain web masing-masing. Pendekatan ini membantu memastikan integritas dan kegunaan data yang dikumpulkan, karena terorganisir dengan rapi dan terkait dengan perusahaan dan domain yang sesuai.

Membangun sistem perayapan web yang dapat diskalakan AWS

Bagian ini menjelaskan cara membuat perayap web yang dijelaskan di Arsitektur bagian ini. Ini mencakup pendekatan sistematis untuk menciptakan kumpulan data perusahaan yang kuat dan properti web terkait mereka. Dataset ini berfungsi sebagai dasar untuk aktivitas crawling Anda. Kemudian, bagian ini menjelaskan cara membangun web crawler etis diPython.

Topik

Mempersiapkan dataset

Jika Anda belum melakukannya, siapkan kumpulan data terperinci dari situs web yang ingin Anda kumpulkan informasinya. Dataset ini harus menyertakan nama domain URL situs web dan nama subdomain yang relevan. Bagian ini menyediakan step-by-step proses untuk membangun kumpulan data ini.

Untuk menyiapkan dataset

1. Tentukan ruang lingkup — Tentukan industri atau sektor yang Anda fokuskan. Tentukan berapa banyak perusahaan yang akan dimasukkan. Dan tentukan kriteria apa pun yang ingin Anda kumpulkan tentang perusahaan-perusahaan ini, seperti jumlah karyawan, lokasi, atau pendapatan.
2. Identifikasi sumber data — Identifikasi sumber informasi apa yang dapat Anda gunakan untuk mengumpulkan informasi tentang perusahaan-perusahaan ini. Contohnya termasuk direktori bisnis (seperti Crunchbase, Bloomberg, atau Forbes), bursa saham (seperti NYSE dan NASDAQ), asosiasi atau publikasi khusus industri, atau database pemerintah (seperti pengajuan SEC).
3. Buat tabel — Di alat pilihan Anda, seperti Microsoft Excel, Google Spreadsheet, atau sistem manajemen database, buat tabel untuk mengumpulkan kriteria tentang setiap perusahaan.

Sertakan kolom untuk setiap kriteria. Minimal, sertakan kolom untuk nama perusahaan, domain utama, subdomain, industri, ukuran, dan lokasi.

4. Kumpulkan informasi awal perusahaan — Kumpulkan informasi berikut tentang masing-masing perusahaan dan masukkan ke dalam tabel yang Anda buat:
   - Nama perusahaan
   - Industri atau sektor
   - Ukuran perusahaan (jumlah karyawan)
   - Pendapatan
   - Lokasi kantor pusat perusahaan
5. Kumpulkan informasi domain — Untuk setiap perusahaan, ekstrak nama domain utama dari URL situs web utama, seperti`example.com`. Anda dapat memverifikasi informasi domain dengan menggunakan alat pencarian domain WHOIS.
6. Kumpulkan informasi subdomain — Untuk setiap perusahaan, teliti subdomain terdaftar, seperti. `blog.example.com` Anda dapat menggunakan alat enumerasi subdomain, seperti Sublist3r, OWASP Amass, atau Subfinder. Anda dapat melakukan Google dorking (dengan mencari`site:example.com`), memeriksa catatan DNS dengan menggunakan `dig` perintah atau alat pencarian DNS, atau Anda dapat menganalisis sertifikat SSL atau TLS.
7. Validasi dan bersihkan data — Tinjau, verifikasi, dan standarisasi data yang telah Anda kumpulkan. Misalnya, hapus entri duplikat, hapus informasi URL yang tidak perlu dari domain dan subdomain, dan verifikasi bahwa semua domain dan subdomain aktif.
8. (Opsional) Kategorikan subdomain — Anda dapat mengkategorikan subdomain ke dalam tipe. Berikut ini adalah beberapa contoh kategori yang mungkin Anda temui:
   - Blog, seperti `blog.example.com`
   - Support atau bantuan, seperti `support.example.com` atau `help.example.com`
   - E-commerce, seperti `shop.example.com` atau `store.example.com`
   - Sumber daya pengembang, seperti `dev.example.com` atau `api.example.com`
   - Daerah atau lokasi, seperti `us.example.com` atau `uk.example.com`
9. (Opsional) Tambahkan metadata yang relevan — Anda dapat merekam metadata yang relevan dalam kumpulan data. Misalnya, Anda dapat menambahkan tanggal terakhir yang diperbarui, sumber informasi, atau skor kepercayaan diri Anda untuk akurasi subdomain.
10. Menerapkan kontrol versi — Gunakan sistem kontrol versi, seperti Git, untuk melacak perubahan pada tabel dari waktu ke waktu. Cadangkan dataset secara teratur.

11. Pertahankan tabel — Siapkan jadwal, seperti triwulanan, untuk memperbarui tabel. Standarisasi dan terapkan proses untuk menambahkan perusahaan baru atau menghapus yang tidak lagi Anda butuhkan. Jika memungkinkan, otomatiskan penemuan subdomain.

Membangun web crawler

Seperti yang dijelaskan di Arsitektur bagian ini, aplikasi berjalan dalam batch — satu untuk setiap perusahaan.

Topik

- Menangkap dan memproses file robots.txt
- Menangkap dan memproses peta situs
- Merancang crawler

Menangkap dan memproses file robots.txt

Setelah menyiapkan kumpulan data, Anda perlu mengonfirmasi apakah domain tersebut memiliki file robots.txt. Untuk perayap web dan bot lainnya, file robots.txt menunjukkan bagian situs web mana yang diizinkan untuk dikunjungi. Menghormati instruksi dalam file ini adalah praktik terbaik yang penting untuk merayapi situs web secara etis. Untuk informasi selengkapnya, lihat Praktik terbaik untuk perayap web etis dalam panduan ini.

Untuk menangkap dan memproses file robots.txt

1. Jika Anda belum melakukannya, instal `requests` pustaka dengan menjalankan perintah berikut di terminal:

```
pip install requests
```

2. Jalankan skrip berikut. Skrip ini melakukan hal berikut:

   - Ini mendefinisikan `check_robots_txt` fungsi yang mengambil domain sebagai input.
   - Ini membangun URL lengkap untuk file robots.txt.
   - Ini mengirimkan permintaan GET ke URL untuk file robots.txt.
   - Jika permintaan berhasil (kode status 200), maka file robots.txt ada.

- Jika permintaan gagal atau mengembalikan kode status yang berbeda, maka file robots.txt tidak ada atau tidak dapat diakses.

```
import requests
from urllib.parse import urljoin
def check_robots_txt(domain):
    # Ensure the domain starts with a protocol
    if not domain.startswith(('http://', 'https://')):
        domain = 'https://' + domain
    # Construct the full URL for robots.txt
    robots_url = urljoin(domain, '/robots.txt')
    try:
        # Send a GET request to the robots.txt URL
        response = requests.get(robots_url, timeout=5)
        # Check if the request was successful (status code 200)
        if response.status_code == 200:
            print(f"robots.txt found at {robots_url}")
            return True
        else:
            print(f"No robots.txt found at {robots_url} (Status code:
 {response.status_code})")
            return False
    except requests.RequestException as e:
        print(f"Error checking {robots_url}: {e}")
        return False
```

> **Note**
>
> Skrip ini menangani pengecualian untuk kesalahan jaringan atau masalah lainnya.

3. Jika file robots.txt ada, gunakan skrip berikut untuk mengunduhnya:

```
import requests

def download(self, url):
    response = requests.get(url, headers=self.headers, timeout=5)
    response.raise_for_status()  # Raise an exception for non-2xx responses
    return response.text

def download_robots_txt(self):
```

```
    # Append '/robots.txt' to the URL to get the robots.txt file's URL
    robots_url = self.url.rstrip('/') + '/robots.txt'
    try:
        response = download(robots_url)
        return response
    except requests.exceptions.RequestException as e:
        print(f"Error downloading robots.txt: {e}, \nGenerating sitemap using
combinations...")
        return e
```

> **Note**
>
> Skrip ini dapat disesuaikan atau dimodifikasi sesuai dengan kasus penggunaan Anda. Anda juga dapat menggabungkan skrip ini.

Menangkap dan memproses peta situs

Selanjutnya, Anda perlu memproses peta situs. Anda dapat menggunakan peta situs untuk memfokuskan crawling pada halaman penting. Ini meningkatkan efisiensi perayapan. Untuk informasi selengkapnya, lihat Praktik terbaik untuk perayap web etis dalam panduan ini.

Untuk menangkap dan memproses peta situs

- Jalankan skrip berikut. Skrip ini mendefinisikan `check_and_download_sitemap` fungsi yang:
  - Menerima URL dasar, URL peta situs opsional dari robots.txt, dan string user-agent.
  - Memeriksa beberapa lokasi peta situs potensial, termasuk yang dari robots.txt (jika disediakan).
  - Mencoba mengunduh peta situs dari setiap lokasi.
  - Memverifikasi bahwa konten yang diunduh dalam format XHTML.
  - Memanggil `parse_sitemap` fungsi untuk mengekstrak file URLs. Fungsi ini:
    - Mem-parsing konten XHTML dari sitemap.
    - Menangani peta situs reguler dan file indeks peta situs.
    - Mengambil sub-peta situs secara rekursif jika indeks peta situs ditemukan.

```
import requests
```

```
from urllib.parse import urljoin
import xml.etree.ElementTree as ET

def check_and_download_sitemap(base_url, robots_sitemap_url=None,
 user_agent='SitemapBot/1.0'):
    headers = {'User-Agent': user_agent}
    sitemap_locations = [robots_sitemap_url, urljoin(base_url, '/sitemap.xml'),
 urljoin(base_url, '/sitemap_index.xml'),
        urljoin(base_url, '/sitemap/'), urljoin(base_url, '/sitemap/sitemap.xml')]

    for sitemap_url in sitemap_locations:
        if not sitemap_url:
            continue
        print(f"Checking for sitemap at: {sitemap_url}")
        try:
            response = requests.get(sitemap_url, headers=headers, timeout=10)
            if response.status_code == 200:
                content_type = response.headers.get('Content-Type', '')
                if 'xml' in content_type:
                    print(f"Successfully downloaded sitemap from {sitemap_url}")
                    return parse_sitemap(response.text)
                else:
                    print(f"Found content at {sitemap_url}, but it's not XML.
 Content-Type: {content_type}")
        except requests.RequestException as e:
            print(f"Error downloading sitemap from {sitemap_url}: {e}")
    print("No sitemap found.")
    return []

def parse_sitemap(sitemap_content):
    urls = []
    try:
        root = ET.fromstring(sitemap_content)
        # Handle both sitemap and sitemapindex
        for loc in root.findall('.//{http://www.sitemaps.org/schemas/
sitemap/0.9}loc'):
            urls.append(loc.text)

        # If it's a sitemap index, recursively fetch each sitemap
        if root.tag.endswith('sitemapindex'):
            all_urls = []
            for url in urls:
                print(f"Fetching sub-sitemap: {url}")
```

```
                sub_sitemap_urls = check_and_download_sitemap(url)
                all_urls.extend(sub_sitemap_urls)
            return all_urls
    except ET.ParseError as e:
        print(f"Error parsing sitemap XML: {e}")
    return urls


if __name__ == "__main__":
    base_url = input("Enter the base URL of the website: ")
    robots_sitemap_url = input("Enter the sitemap URL from robots.txt (or press
 Enter if none): ").strip() or None
    urls = check_and_download_sitemap(base_url, robots_sitemap_url)
    print(f"Found {len(urls)} URLs in sitemap:")
    for url in urls[:5]:  # Print first 5 URLs as an example
        print(url)
    if len(urls) > 5:
        print("...")
```

Merancang crawler

Selanjutnya, Anda mendesain web crawler. Crawler dirancang untuk mengikuti praktik terbaik yang dijelaskan Praktik terbaik untuk perayap web etis dalam panduan ini. `EthicalCrawler`Kelas ini menunjukkan beberapa prinsip kunci merangkak etis:

- Mengambil dan mengurai file robots.txt - Crawler mengambil file robots.txt untuk situs web target.
- Menghormati izin perayapan — Sebelum merayapi URL apa pun, crawler memeriksa apakah aturan dalam file robots.txt memungkinkan crawling untuk URL tersebut. Jika URL tidak diizinkan, maka crawler melewatinya dan pindah ke URL berikutnya.
- Menghormati penundaan crawl - Crawler memeriksa direktif penundaan perayapan di file robots.txt. Jika satu ditentukan, crawler menggunakan penundaan ini di antara permintaan. Jika tidak, ia menggunakan penundaan default.
- Identifikasi agen pengguna - Crawler menggunakan string agen pengguna khusus untuk mengidentifikasi dirinya ke situs web. Jika diperlukan, pemilik situs web dapat menetapkan aturan khusus untuk membatasi atau mengizinkan crawler Anda.
- Penanganan kesalahan dan degradasi anggun - Jika file robots.txt tidak dapat diambil atau diurai, crawler melanjutkan dengan aturan default konservatif. Ini menangani kesalahan jaringan dan respons HTTP non-200.

- Perayapan terbatas — Untuk menghindari kewalahan server, ada batasan berapa banyak halaman yang dapat dirayapi.

Skrip berikut adalah pseudocode yang menjelaskan cara kerja crawler web:

```
import requests
from urllib.parse import urljoin, urlparse
import time

class EthicalCrawler:
    def __init__(self, start_url, user_agent='EthicalBot/1.0'):
        self.start_url = start_url
        self.user_agent = user_agent
        self.domain = urlparse(start_url).netloc
        self.robots_parser = None
        self.crawl_delay = 1  # Default delay in seconds

    def can_fetch(self, url):
        if self.robots_parser:
            return self.robots_parser.allowed(url, self.user_agent)
        return True  # If no robots.txt, assume allowed but crawl conservatively

    def get_crawl_delay(self):
        if self.robots_parser:
            delay = self.robots_parser.agent(self.user_agent).delay
            if delay is not None:
                self.crawl_delay = delay
        print(f"Using crawl delay of {self.crawl_delay} seconds")

    def crawl(self, max_pages=10):
        self.get_crawl_delay()
        pages_crawled = 0
        urls_to_crawl = [self.start_url]
        while urls_to_crawl and pages_crawled < max_pages:
            url = urls_to_crawl.pop(0)
            if not self.can_fetch(url):
                print(f"robots.txt disallows crawling: {url}")
                continue
            try:
                response = requests.get(url, headers={'User-Agent': self.user_agent})
                if response.status_code == 200:
                    print(f"Successfully crawled: {url}")
```

```
                # Here you would typically parse the content, extract links, etc.
                # For this example, we'll just increment the counter
                pages_crawled += 1
            else:
                print(f"Failed to crawl {url}: HTTP {response.status_code}")
        except Exception as e:
            print(f"Error crawling {url}: {e}")

        # Respect the crawl delay
        time.sleep(self.crawl_delay)

    print(f"Crawling complete. Crawled {pages_crawled} pages.")
```

Untuk membangun perayap web canggih dan etis yang mengumpulkan data ESG

1. Salin contoh kode berikut untuk crawler web etis tingkat lanjut yang digunakan dalam sistem ini:

```
import requests
from urllib.parse import urljoin, urlparse
import time
from collections import deque
import random
from bs4 import BeautifulSoup
import re
import csv
import os


class EnhancedESGCrawler:
    def __init__(self, start_url):
        self.start_url = start_url
        self.domain = urlparse(start_url).netloc
        self.desktop_user_agent = 'ESGEthicalBot/1.0'
        self.mobile_user_agent = 'Mozilla/5.0 (iPhone; CPU iPhone OS 14_0 like
 Mac OS X) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/14.0 Mobile/15E148
 Safari/604.1'
        self.robots_parser = None
        self.crawl_delay = None
        self.urls_to_crawl = deque()
        self.crawled_urls = set()
        self.max_retries = 2
        self.session = requests.Session()
        self.esg_data = []
```

```
        self.news_links = []
        self.pdf_links = []

    def setup(self):
        self.fetch_robots_txt() # Provided in Previous Snippet
        self.fetch_sitemap() # Provided in Previous Snippet

    def can_fetch(self, url, user_agent):
        if self.robots_parser:
            return self.robots_parser.allowed(url, user_agent)
        return True

    def delay(self):
        if self.crawl_delay is not None:
            time.sleep(self.crawl_delay)
        else:
            time.sleep(random.uniform(1, 3))

    def get_headers(self, user_agent):
        return {'User-Agent': user_agent,
                'Accept': 'text/html,application/xhtml+xml,application/
xml;q=0.9,image/webp,*/*;q=0.8',
                'Accept-Language': 'en-US,en;q=0.5', 'Accept-Encoding': 'gzip,
 deflate, br', 'DNT': '1',
                'Connection': 'keep-alive', 'Upgrade-Insecure-Requests': '1'}

    def extract_esg_data(self, url, html_content):
        soup = BeautifulSoup(html_content, 'html.parser')
        esg_data = {
            'url': url,
            'environmental': self.extract_environmental_data(soup),
            'social': self.extract_social_data(soup),
            'governance': self.extract_governance_data(soup)
        }
        self.esg_data.append(esg_data)
        # Extract news links and PDFs
        self.extract_news_links(soup, url)
        self.extract_pdf_links(soup, url)

    def extract_environmental_data(self, soup):
        keywords = ['carbon footprint', 'emissions', 'renewable energy', 'waste
 management', 'climate change']
        return self.extract_keyword_data(soup, keywords)
```

```
   def extract_social_data(self, soup):
       keywords = ['diversity', 'inclusion', 'human rights', 'labor practices',
'community engagement']
       return self.extract_keyword_data(soup, keywords)

   def extract_governance_data(self, soup):
       keywords = ['board structure', 'executive compensation', 'shareholder
rights', 'ethics', 'transparency']
       return self.extract_keyword_data(soup, keywords)

   def extract_keyword_data(self, soup, keywords):
       text = soup.get_text().lower()
       return {keyword: len(re.findall(r'\b' + re.escape(keyword) + r'\b', text))
for keyword in keywords}

   def extract_news_links(self, soup, base_url):
       news_keywords = ['news', 'press release', 'article', 'blog',
'sustainability']
       for a in soup.find_all('a', href=True):
           if any(keyword in a.text.lower() for keyword in news_keywords):
               full_url = urljoin(base_url, a['href'])
               if full_url not in self.news_links:
                   self.news_links.append({'url': full_url, 'text':
a.text.strip()})

   def extract_pdf_links(self, soup, base_url):
       for a in soup.find_all('a', href=True):
           if a['href'].lower().endswith('.pdf'):
               full_url = urljoin(base_url, a['href'])
               if full_url not in self.pdf_links:
                   self.pdf_links.append({'url': full_url, 'text':
a.text.strip()})

   def is_relevant_to_sustainable_finance(self, text):
       keywords = ['sustainable finance', 'esg', 'green bond', 'social impact',
'environmental impact',
                   'climate risk', 'sustainability report', 'corporate
responsibility']
       return any(keyword in text.lower() for keyword in keywords)

   def attempt_crawl(self, url, user_agent):
       for _ in range(self.max_retries):
```

```
            try:
                response = self.session.get(url,
 headers=self.get_headers(user_agent), timeout=10)
                if response.status_code == 200:
                    print(f"Successfully crawled: {url}")
                    if response.headers.get('Content-Type', '').startswith('text/
html'):
                        self.extract_esg_data(url, response.text)
                    elif response.headers.get('Content-Type',
 '').startswith('application/pdf'):
                        self.save_pdf(url, response.content)
                    return True
                else:
                    print(f"Failed to crawl {url}: HTTP {response.status_code}")
            except requests.RequestException as e:
                print(f"Error crawling {url} with {user_agent}: {e}")

            self.delay()
        return False

    def crawl_url(self, url):
        if not self.can_fetch(url, self.desktop_user_agent):
            print(f"Robots.txt disallows desktop user agent: {url}")
            if self.can_fetch(url, self.mobile_user_agent):
                print(f"Attempting with mobile user agent: {url}")
                return self.attempt_crawl(url, self.mobile_user_agent)
            else:
                print(f"Robots.txt disallows both user agents: {url}")
                return False

        return self.attempt_crawl(url, self.desktop_user_agent)

    def crawl(self, max_pages=100):
        self.setup()

        if not self.urls_to_crawl:
            self.urls_to_crawl.append(self.start_url)

        pages_crawled = 0
        while self.urls_to_crawl and pages_crawled < max_pages:
            url = self.urls_to_crawl.popleft()
            if url not in self.crawled_urls:
                if self.crawl_url(url):
```

```
                    pages_crawled += 1
                self.crawled_urls.add(url)
                self.delay()

        print(f"Crawling complete. Successfully crawled {pages_crawled} pages.")
        self.save_esg_data()
        self.save_news_links()
        self.save_pdf_links()

    def save_esg_data(self):
        with open('esg_data.csv', 'w', newline='', encoding='utf-8') as file:
            writer = csv.DictWriter(file, fieldnames=['url', 'environmental',
'social', 'governance'])
            writer.writeheader()
            for data in self.esg_data:
                writer.writerow({
                    'url': data['url'],
                    'environmental': ', '.join([f"{k}: {v}" for k, v in
data['environmental'].items()]),
                    'social': ', '.join([f"{k}: {v}" for k, v in
data['social'].items()]),
                    'governance': ', '.join([f"{k}: {v}" for k, v in
data['governance'].items()])
                })
        print("ESG data saved to esg_data.csv")

    def save_news_links(self):
        with open('news_links.csv', 'w', newline='', encoding='utf-8') as file:
            writer = csv.DictWriter(file, fieldnames=['url', 'text', 'relevant'])
            writer.writeheader()
            for news in self.news_links:
                writer.writerow({
                    'url': news['url'],
                    'text': news['text'],
                    'relevant':
self.is_relevant_to_sustainable_finance(news['text'])
                })
        print("News links saved to news_links.csv")

    def save_pdf_links(self):
        # Code for saving PDF in S3 or filesystem

    def save_pdf(self, url, content):
```

```
        # Code for saving PDF in S3 or filesystem

# Example usage
if __name__ == "__main__":
    start_url = input("Enter the starting URL to crawl for ESG data and news: ")
    crawler = EnhancedESGCrawler(start_url)
    crawler.crawl(max_pages=50)
```

2. Siapkan berbagai atribut, termasuk agen pengguna, koleksi kosong untuk URLs, dan daftar penyimpanan data.
3. Sesuaikan kata kunci dan kriteria relevansi dalam `is_relevant_to_sustainable_finance()` metode agar sesuai dengan kebutuhan spesifik Anda.
4. Pastikan file robots.txt mengizinkan perayapan situs web dan Anda menggunakan penundaan perayapan dan agen pengguna yang ditentukan dalam file robots.txt.
5. Pertimbangkan untuk membuat penyesuaian berikut ke skrip perayap web yang disediakan, sesuai kebutuhan untuk organisasi Anda:

    - Menerapkan `fetch_sitemap()` metode untuk penemuan URL yang lebih efisien.
    - Tingkatkan pencatatan kesalahan dan penanganan untuk penggunaan produksi.
    - Menerapkan analisis relevansi konten yang lebih canggih.
    - Tambahkan kontrol kedalaman dan lebar untuk membatasi cakupan perayapan.

Membangun AWS infrastruktur

Ada banyak Layanan AWS yang dapat Anda gunakan untuk membangun infrastruktur web crawling. Bagian Arsitektur dari panduan ini mencakup satu solusi yang diusulkan. Kami menyarankan Anda mempertimbangkan untuk menggunakan yang berikut ini Layanan AWS untuk membangun infrastruktur pendukung untuk crawler web Anda:

- Gunakan Amazon Virtual Private Cloud (Amazon VPC) untuk membuat VPC dan subnet.
- Memulai proses crawling dengan menggunakan Amazon EventBridge Scheduler.
- Kelola pekerjaan crawler web dengan menggunakan AWS Batch pekerjaan dan antrian pekerjaan.
- Gunakan salah satu solusi berikut untuk menjalankan pekerjaan crawler web:
    - Kontainer Amazon Elastic Container Service (Amazon ECS) aktif AWS Fargate
    - Instans Amazon Elastic Compute Cloud (Amazon EC2)

> **Note**
>
> Jika aplikasi Anda dapat menangani gangguan, pertimbangkan untuk menggunakan Instans Spot Amazon EC2 melalui Armada Spot. Armada Instans Spot dapat membantu Anda menghemat biaya komputasi secara signifikan.

  - AWS Lambda fungsi
- Simpan data yang diambil dan file mentah dalam bucket Amazon Simple Storage Service (Amazon S3).

Praktik terbaik untuk perayap web etis

Bagian ini membahas praktik terbaik dan pertimbangan etis utama untuk membangun aplikasi perayapan web yang mengumpulkan data lingkungan, sosial, dan tata kelola (ESG). Dengan mengikuti praktik terbaik ini, Anda dapat melindungi proyek dan organisasi Anda dan berkontribusi pada ekosistem web yang lebih bertanggung jawab dan berkelanjutan. Pendekatan ini membantu Anda mengakses data berharga dan menggunakannya untuk penelitian, bisnis, dan inovasi dengan cara yang menghormati semua pemangku kepentingan.

Kepatuhan Robots.txt

File robots.txt digunakan di situs web untuk berkomunikasi dengan perayap web dan bot tentang bagian mana dari situs web yang harus atau tidak boleh diakses atau dirayapi. Saat perayap web menemukan file robots.txt di situs web, perayap web mem-parsing instruksi dan menyesuaikan perilaku perayapannya. Ini mencegah crawler melanggar instruksi pemilik situs web dan menjaga hubungan kerja sama antara situs web dan crawler. Oleh karena itu, file robots.txt membantu dengan kontrol akses, perlindungan konten sensitif, manajemen beban, dan kepatuhan hukum.

Sebaiknya Anda untuk mematuhi praktik terbaik berikut:

- Selalu periksa dan hormati aturan dalam file robots.txt.
- Sebelum merayapi URL apa pun, periksa aturan untuk agen pengguna desktop dan seluler.
- Jika situs web hanya mengizinkan agen pengguna seluler, gunakan header agen yang berbeda, seperti header agen seluler, untuk permintaan Anda.

Tidak adanya file robots.txt tidak selalu berarti Anda tidak dapat atau tidak boleh merayapi situs web. Crawling harus selalu dilakukan secara bertanggung jawab, menghormati sumber daya situs web dan hak implisit pemilik. Berikut ini adalah praktik terbaik yang direkomendasikan ketika robots.txt tidak ada:

- Asumsikan crawling diperbolehkan, tetapi lanjutkan dengan hati-hati.
- Menerapkan praktik perayapan yang sopan.
- Pertimbangkan untuk menghubungi pemilik situs web untuk mendapatkan izin jika Anda berencana melakukan crawling ekstensif.

Pembatasan laju perayapan

Gunakan kecepatan perayapan yang wajar untuk menghindari kewalahan server. Menerapkan penundaan antar permintaan, baik seperti yang ditentukan oleh file robots.txt atau dengan menggunakan penundaan acak. Untuk situs web berukuran kecil atau menengah, 1 permintaan setiap 10-15 detik mungkin sesuai. Untuk situs web yang lebih besar atau yang memiliki izin perayapan eksplisit, 1-2 permintaan per detik mungkin sesuai.

Transparansi agen pengguna

Identifikasi crawler Anda di header user-agent. Informasi header HTTP ini dimaksudkan untuk mengidentifikasi perangkat yang meminta konten. Umumnya, kata bot termasuk dalam nama agen. Crawler dan bot lainnya terkadang menggunakan bidang penting di header untuk menyertakan informasi kontak.

Perayapan yang efisien

Gunakan peta situs, yang dikembangkan oleh pemilik situs web, untuk fokus pada halaman penting.

Pendekatan adaptif

Program crawler untuk beralih ke agen pengguna seluler jika versi desktop tidak berhasil. Ini dapat memberikan akses crawler dan mengurangi ketegangan pada server situs web.

Penanganan kesalahan

Pastikan crawler menangani berbagai kode status HTTP dengan tepat. Misalnya, crawler harus berhenti sejenak jika menemukan kode status 429 (“Terlalu banyak permintaan”). Jika crawler terus menerima 403 kode status (“Terlarang”), maka pertimbangkan untuk menghentikan crawling.

Merangkak dalam batch

Sebaiknya Anda melakukan tindakan berikut:

- Alih-alih merangkak URLs sekaligus, bagilah tugas menjadi batch yang lebih kecil. Ini dapat membantu mendistribusikan beban dan mengurangi risiko menghadapi masalah, seperti batas waktu atau kendala sumber daya.

- Jika keseluruhan tugas perayapan diharapkan berjalan lama, pertimbangkan untuk membaginya menjadi beberapa tugas yang lebih kecil dan lebih mudah dikelola. Ini dapat membuat proses lebih terukur dan tangguh.
- Jika jumlah crawl relatif kecil, pertimbangkan URLs untuk menggunakan solusi tanpa server, seperti. AWS Lambda Fungsi Lambda dapat cocok untuk tugas yang berumur pendek dan digerakkan oleh peristiwa karena mereka secara otomatis menskalakan dan menangani manajemen sumber daya.

Keamanan

Untuk tugas komputasi perayapan web, sebaiknya Anda mengonfigurasi lingkungan agar hanya mengizinkan lalu lintas keluar. Ini membantu meningkatkan keamanan dengan meminimalkan permukaan serangan dan mengurangi risiko akses masuk yang tidak sah. Mengizinkan hanya koneksi keluar memungkinkan proses perayapan untuk berkomunikasi dengan situs web target dan mengambil data yang diperlukan, dan membatasi lalu lintas masuk yang berpotensi membahayakan sistem.

Pertimbangan lainnya

Tinjau pertimbangan tambahan dan praktik terbaik berikut:

- Periksa pedoman perayapan dalam persyaratan layanan atau kebijakan privasi situs web.
- Cari `meta` tag dalam HTML yang mungkin menyediakan arahan crawling.
- Waspadai pembatasan hukum di yurisdiksi Anda terkait pengumpulan dan penggunaan data.
- Bersiaplah untuk berhenti merangkak jika diminta oleh pemilik situs web.

Pertanyaan yang Sering Diajukan

Bagaimana jika file robots.txt tidak tersedia?

Tidak adanya file robots.txt tidak selalu berarti Anda tidak dapat atau tidak boleh merayapi situs web. Crawling harus selalu dilakukan secara bertanggung jawab, menghormati sumber daya situs web dan hak implisit pemilik situs web.

Bagaimana jika file sitemaps.xml tidak tersedia?

Tergantung pada kebutuhan, Anda dapat melakukan salah satu dari yang berikut:

- Cari peta situs HTML — Cari halaman peta situs HTML yang mencantumkan halaman penting di situs web. Ini sering ditautkan di footer.
- Merayapi dari beranda - Mulai merangkak dari beranda dan ikuti tautan internal untuk menemukan halaman lain.
- Analisis pola URL — Analisis struktur URL situs web untuk mengidentifikasi pola dan menghasilkan potensi secara terprogram. URLs
- Tinjau file robots.txt - Periksa file robots.txt untuk halaman atau direktori yang tidak diizinkan. Ini dapat memberikan petunjuk tentang struktur situs.
- Tinjau titik akhir API — Beberapa situs web menawarkan titik akhir API yang dapat digunakan untuk mengambil konten dan informasi struktur.
- Periksa hasil mesin pencari — Gunakan mesin pencari untuk menemukan halaman situs web yang diindeks dengan menggunakan situs: operator pencarian, seperti. `site:example.com`
- Analisis backlink — Analisis backlink ke situs web untuk menemukan halaman penting yang ditautkan oleh situs lain.
- Tinjau arsip web — Periksa arsip internet, seperti Mesin Wayback, untuk versi situs yang lebih lama yang mungkin memiliki peta situs atau struktur yang berbeda.
- Cari pola sistem manajemen konten (CMS) — Jika Anda dapat mengidentifikasi CMS, gunakan pola URL umum yang terkait dengan sistem itu.
- Konfirmasi JavaScript rendering — Jika situs sangat bergantung JavaScript, pastikan crawler Anda dapat merender JavaScript untuk menemukan konten yang dimuat secara dinamis. Untuk beberapa situs web, file sitemap.xml dimuat setelah JavaScript rendering diaktifkan.

Dapatkah saya menggunakan solusi tanpa server alih-alih Amazon EC2 atau Amazon ECS?

Ya. AWS Lambdafungsi untuk perayapan web dapat menjadi opsi yang layak, terutama untuk tugas perayapan skala kecil atau lebih modular. Namun, untuk operasi crawling skala besar yang berjalan lama, pendekatan yang lebih tradisional yang menggunakan instans Amazon Elastic Compute Cloud (Amazon EC2) atau Amazon Elastic Container Service (Amazon ECS) mungkin lebih cocok. Penting untuk mengevaluasi dengan cermat persyaratan dan trade-off spesifik Anda saat memilih layanan komputasi yang tepat untuk kebutuhan crawling web Anda.

Mengapa crawler mendapatkan kode status 403?

HTTP 403 adalah kode status HTTP yang berarti akses ke sumber daya yang diminta dilarang. Jika permintaan itu benar, maka server memahami permintaan dan tidak akan memenuhinya. Untuk mencegah kode status 403, Anda dapat melakukan hal berikut:

- Batasi kecepatan crawl Anda.
- Periksa apakah peta situs atau file robots.txt memungkinkan crawler mengakses URL.
- Coba dengan agen pengguna seluler alih-alih agen pengguna desktop.

Jika tidak ada yang berhasil di atas, Anda harus menghormati keputusan pemilik situs web dan tidak merayapi halaman.

Langkah dan sumber daya selanjutnya

Setelah mengumpulkan data lingkungan, sosial, dan tata kelola (ESG) mentah, Anda dapat melakukan hal berikut untuk mengekstrak informasi yang bermakna dari data:

1. Bersihkan data — Data mungkin mencakup sejumlah besar informasi yang tidak relevan yang tidak terkait dengan faktor ESG dan data keuangan. Penting untuk menghapus data yang tidak relevan ini dan hanya menyimpan informasi yang diperlukan untuk melakukan analisis yang diperlukan. Anda dapat menggunakan alat, seperti yfinance, untuk membantu membersihkan data.
2. Ekstrak dan ubah data — Ekstrak fitur atau variabel yang relevan dari data mentah dan ubah menjadi format yang cocok untuk analisis. Anda dapat mengubah data menjadi format tabel untuk keterbacaan dan kejelasan yang lebih baik. Anda dapat menggunakan perpustakaan, seperti pandas, untuk memperbaiki data. Anda juga dapat menggunakan rekayasa fitur, normalisasi data, dan metrik turunan untuk mengubah data.
3. Lakukan analitik — Anda dapat melakukan berbagai tugas analitis. Ini mungkin termasuk menghasilkan statistik deskriptif, membuat visualisasi data, dan melakukan analisis data eksplorasi untuk mendapatkan wawasan tentang kinerja ESG perusahaan.
4. Terapkan pembelajaran mesin — Anda dapat menggunakan data yang dibersihkan dan diubah untuk melatih model pembelajaran mesin. Model-model ini dapat membantu Anda mengidentifikasi perusahaan yang saat ini menunjukkan keberlanjutan keuangan dan memproyeksikan kinerja keberlanjutan masa depan mereka.

Dengan menggunakan perayap web dan proses evaluasi data ini, Anda dapat secara efektif memperoleh pemahaman komprehensif tentang praktik keberlanjutan dan kinerja keuangan perusahaan yang Anda evaluasi. Anda dapat menggunakan informasi ini untuk menginformasikan keputusan investasi, melacak kemajuan, dan mendukung praktik bisnis yang berkelanjutan.

Sumber daya

- Apa itu perayap web? (Cloudflare situs web)
- Panduan untuk investasi ESG (Investopedia situs web)

Alat

- Beautiful Soup (Beautiful Soup dokumentasi)
- Menangani tabular, data relasional (pandas situs web)

Riwayat dokumen

Tabel berikut menjelaskan perubahan signifikan pada panduan ini. Jika Anda ingin diberi tahu tentang pembaruan masa depan, Anda dapat berlangganan umpan RSS.

Perubahan	Deskripsi	Tanggal
Publikasi awal	—	Januari 6, 2025

AWS Glosarium Panduan Preskriptif

Berikut ini adalah istilah yang umum digunakan dalam strategi, panduan, dan pola yang disediakan oleh Panduan AWS Preskriptif. Untuk menyarankan entri, silakan gunakan tautan Berikan umpan balik di akhir glosarium.

Nomor

7 Rs

Tujuh strategi migrasi umum untuk memindahkan aplikasi ke cloud. Strategi ini dibangun di atas 5 Rs yang diidentifikasi Gartner pada tahun 2011 dan terdiri dari yang berikut:

- Refactor/Re-Architect — Memindahkan aplikasi dan memodifikasi arsitekturnya dengan memanfaatkan sepenuhnya fitur cloud-native untuk meningkatkan kelincahan, kinerja, dan skalabilitas. Ini biasanya melibatkan porting sistem operasi dan database. Contoh: Migrasikan database Oracle lokal Anda ke Amazon Aurora PostgreSQL Compatible Edition.
- Replatform (angkat dan bentuk ulang) — Pindahkan aplikasi ke cloud, dan perkenalkan beberapa tingkat pengoptimalan untuk memanfaatkan kemampuan cloud. Contoh: Memigrasikan database Oracle lokal Anda ke Amazon Relational Database Service (Amazon RDS) untuk Oracle di. AWS Cloud
- Pembelian kembali (drop and shop) - Beralih ke produk yang berbeda, biasanya dengan beralih dari lisensi tradisional ke model SaaS. Contoh: Migrasikan sistem manajemen hubungan pelanggan (CRM) Anda ke Salesforce.com.
- Rehost (lift dan shift) — Pindahkan aplikasi ke cloud tanpa membuat perubahan apa pun untuk memanfaatkan kemampuan cloud. Contoh: Migrasikan database Oracle lokal Anda ke Oracle pada instans EC2 di. AWS Cloud
- Relokasi (hypervisor-level lift and shift) — Pindahkan infrastruktur ke cloud tanpa membeli perangkat keras baru, menulis ulang aplikasi, atau memodifikasi operasi yang ada. Anda memigrasikan server dari platform lokal ke layanan cloud untuk platform yang sama. Contoh: Migrasikan Microsoft Hyper-V aplikasi ke AWS.
- Pertahankan (kunjungi kembali) - Simpan aplikasi di lingkungan sumber Anda. Ini mungkin termasuk aplikasi yang memerlukan refactoring besar, dan Anda ingin menunda pekerjaan itu sampai nanti, dan aplikasi lama yang ingin Anda pertahankan, karena tidak ada pembenaran bisnis untuk memigrasikannya.

- Pensiun — Menonaktifkan atau menghapus aplikasi yang tidak lagi diperlukan di lingkungan sumber Anda.

A

ABAC

Lihat kontrol akses berbasis atribut.

layanan abstrak

Lihat layanan terkelola.

ASAM

Lihat atomisitas, konsistensi, isolasi, daya tahan.

migrasi aktif-aktif

Metode migrasi database di mana database sumber dan target tetap sinkron (dengan menggunakan alat replikasi dua arah atau operasi penulisan ganda), dan kedua database menangani transaksi dari menghubungkan aplikasi selama migrasi. Metode ini mendukung migrasi dalam batch kecil yang terkontrol alih-alih memerlukan pemotongan satu kali. Ini lebih fleksibel tetapi membutuhkan lebih banyak pekerjaan daripada migrasi aktif-pasif.

migrasi aktif-pasif

Metode migrasi database di mana database sumber dan target disimpan dalam sinkron, tetapi hanya database sumber yang menangani transaksi dari menghubungkan aplikasi sementara data direplikasi ke database target. Basis data target tidak menerima transaksi apa pun selama migrasi.

fungsi agregat

Fungsi SQL yang beroperasi pada sekelompok baris dan menghitung nilai pengembalian tunggal untuk grup. Contoh fungsi agregat meliputi `SUM` dan`MAX`.

AI

Lihat kecerdasan buatan.

AIOps

Lihat operasi kecerdasan buatan.

anonimisasi

Proses menghapus informasi pribadi secara permanen dalam kumpulan data. Anonimisasi dapat membantu melindungi privasi pribadi. Data anonim tidak lagi dianggap sebagai data pribadi.

anti-pola

Solusi yang sering digunakan untuk masalah berulang di mana solusinya kontra-produktif, tidak efektif, atau kurang efektif daripada alternatif.

kontrol aplikasi

Pendekatan keamanan yang memungkinkan penggunaan hanya aplikasi yang disetujui untuk membantu melindungi sistem dari malware.

portofolio aplikasi

Kumpulan informasi rinci tentang setiap aplikasi yang digunakan oleh organisasi, termasuk biaya untuk membangun dan memelihara aplikasi, dan nilai bisnisnya. Informasi ini adalah kunci untuk penemuan portofolio dan proses analisis dan membantu mengidentifikasi dan memprioritaskan aplikasi yang akan dimigrasi, dimodernisasi, dan dioptimalkan.

kecerdasan buatan (AI)

Bidang ilmu komputer yang didedikasikan untuk menggunakan teknologi komputasi untuk melakukan fungsi kognitif yang biasanya terkait dengan manusia, seperti belajar, memecahkan masalah, dan mengenali pola. Untuk informasi lebih lanjut, lihat Apa itu Kecerdasan Buatan?

operasi kecerdasan buatan (AIOps)

Proses menggunakan teknik pembelajaran mesin untuk memecahkan masalah operasional, mengurangi insiden operasional dan intervensi manusia, dan meningkatkan kualitas layanan. Untuk informasi selengkapnya tentang cara AIOps digunakan dalam strategi AWS migrasi, lihat panduan integrasi operasi.

enkripsi asimetris

Algoritma enkripsi yang menggunakan sepasang kunci, kunci publik untuk enkripsi dan kunci pribadi untuk dekripsi. Anda dapat berbagi kunci publik karena tidak digunakan untuk dekripsi, tetapi akses ke kunci pribadi harus sangat dibatasi.

atomisitas, konsistensi, isolasi, daya tahan (ACID)

Satu set properti perangkat lunak yang menjamin validitas data dan keandalan operasional database, bahkan dalam kasus kesalahan, kegagalan daya, atau masalah lainnya.

kontrol akses berbasis atribut (ABAC)

Praktik membuat izin berbutir halus berdasarkan atribut pengguna, seperti departemen, peran pekerjaan, dan nama tim. Untuk informasi selengkapnya, lihat ABAC untuk AWS dokumentasi AWS Identity and Access Management (IAM).

sumber data otoritatif

Lokasi di mana Anda menyimpan versi utama data, yang dianggap sebagai sumber informasi yang paling dapat diandalkan. Anda dapat menyalin data dari sumber data otoritatif ke lokasi lain untuk tujuan pemrosesan atau modifikasi data, seperti menganonimkan, menyunting, atau membuat nama samaran.

Zona Ketersediaan

Lokasi berbeda di dalam Wilayah AWS yang terisolasi dari kegagalan di Availability Zone lainnya dan menyediakan konektivitas jaringan latensi rendah yang murah ke Availability Zone lainnya di Wilayah yang sama.

AWS Kerangka Adopsi Cloud (AWS CAF)

Kerangka pedoman dan praktik terbaik AWS untuk membantu organisasi mengembangkan rencana yang efisien dan efektif untuk bergerak dengan sukses ke cloud. AWS CAF mengatur panduan ke dalam enam area fokus yang disebut perspektif: bisnis, orang, tata kelola, platform, keamanan, dan operasi. Perspektif bisnis, orang, dan tata kelola fokus pada keterampilan dan proses bisnis; perspektif platform, keamanan, dan operasi fokus pada keterampilan dan proses teknis. Misalnya, perspektif masyarakat menargetkan pemangku kepentingan yang menangani sumber daya manusia (SDM), fungsi kepegawaian, dan manajemen orang. Untuk perspektif ini, AWS CAF memberikan panduan untuk pengembangan, pelatihan, dan komunikasi orang untuk membantu mempersiapkan organisasi untuk adopsi cloud yang sukses. Untuk informasi lebih lanjut, lihat situs web AWS CAF dan whitepaper AWS CAF.

AWS Kerangka Kualifikasi Beban Kerja (AWS WQF)

Alat yang mengevaluasi beban kerja migrasi database, merekomendasikan strategi migrasi, dan memberikan perkiraan kerja. AWS WQF disertakan dengan AWS Schema Conversion Tool ()AWS SCT. Ini menganalisis skema database dan objek kode, kode aplikasi, dependensi, dan karakteristik kinerja, dan memberikan laporan penilaian.

B

bot buruk

Bot yang dimaksudkan untuk mengganggu atau menyebabkan kerugian bagi individu atau organisasi.

BCP

Lihat perencanaan kontinuitas bisnis.

grafik perilaku

Pandangan interaktif yang terpadu tentang perilaku dan interaksi sumber daya dari waktu ke waktu. Anda dapat menggunakan grafik perilaku dengan Amazon Detective untuk memeriksa upaya logon yang gagal, panggilan API yang mencurigakan, dan tindakan serupa. Untuk informasi selengkapnya, lihat Data dalam grafik perilaku di dokumentasi Detektif.

sistem big-endian

Sistem yang menyimpan byte paling signifikan terlebih dahulu. Lihat juga endianness.

klasifikasi biner

Sebuah proses yang memprediksi hasil biner (salah satu dari dua kelas yang mungkin). Misalnya, model ML Anda mungkin perlu memprediksi masalah seperti “Apakah email ini spam atau bukan spam?” atau “Apakah produk ini buku atau mobil?”

filter mekar

Struktur data probabilistik dan efisien memori yang digunakan untuk menguji apakah suatu elemen adalah anggota dari suatu himpunan.

deployment biru/hijau

Strategi penyebaran tempat Anda membuat dua lingkungan yang terpisah namun identik. Anda menjalankan versi aplikasi saat ini di satu lingkungan (biru) dan versi aplikasi baru di lingkungan lain (hijau). Strategi ini membantu Anda dengan cepat memutar kembali dengan dampak minimal.

bot

Aplikasi perangkat lunak yang menjalankan tugas otomatis melalui internet dan mensimulasikan aktivitas atau interaksi manusia. Beberapa bot berguna atau bermanfaat, seperti perayap web yang mengindeks informasi di internet. Beberapa bot lain, yang dikenal sebagai bot buruk, dimaksudkan untuk mengganggu atau membahayakan individu atau organisasi.

botnet

Jaringan bot yang terinfeksi oleh malware dan berada di bawah kendali satu pihak, yang dikenal sebagai bot herder atau operator bot. Botnet adalah mekanisme paling terkenal untuk skala bot dan dampaknya.

cabang

Area berisi repositori kode. Cabang pertama yang dibuat dalam repositori adalah cabang utama. Anda dapat membuat cabang baru dari cabang yang ada, dan Anda kemudian dapat mengembangkan fitur atau memperbaiki bug di cabang baru. Cabang yang Anda buat untuk membangun fitur biasanya disebut sebagai cabang fitur. Saat fitur siap dirilis, Anda menggabungkan cabang fitur kembali ke cabang utama. Untuk informasi selengkapnya, lihat Tentang cabang (GitHub dokumentasi).

akses break-glass

Dalam keadaan luar biasa dan melalui proses yang disetujui, cara cepat bagi pengguna untuk mendapatkan akses ke Akun AWS yang biasanya tidak memiliki izin untuk mengaksesnya. Untuk informasi lebih lanjut, lihat indikator Implementasikan prosedur break-glass dalam panduan Well-Architected AWS .

strategi brownfield

Infrastruktur yang ada di lingkungan Anda. Saat mengadopsi strategi brownfield untuk arsitektur sistem, Anda merancang arsitektur di sekitar kendala sistem dan infrastruktur saat ini. Jika Anda memperluas infrastruktur yang ada, Anda dapat memadukan strategi brownfield dan greenfield.

cache penyangga

Area memori tempat data yang paling sering diakses disimpan.

kemampuan bisnis

Apa yang dilakukan bisnis untuk menghasilkan nilai (misalnya, penjualan, layanan pelanggan, atau pemasaran). Arsitektur layanan mikro dan keputusan pengembangan dapat didorong oleh kemampuan bisnis. Untuk informasi selengkapnya, lihat bagian Terorganisir di sekitar kemampuan bisnis dari Menjalankan layanan mikro kontainer di whitepaper. AWS

perencanaan kelangsungan bisnis (BCP)

Rencana yang membahas dampak potensial dari peristiwa yang mengganggu, seperti migrasi skala besar, pada operasi dan memungkinkan bisnis untuk melanjutkan operasi dengan cepat.

C

KAFE

Lihat Kerangka Adopsi AWS Cloud.

penyebaran kenari

Rilis versi yang lambat dan bertahap untuk pengguna akhir. Ketika Anda yakin, Anda menyebarkan versi baru dan mengganti versi saat ini secara keseluruhan.

CCoE

Lihat Cloud Center of Excellence.

CDC

Lihat mengubah pengambilan data.

ubah pengambilan data (CDC)

Proses melacak perubahan ke sumber data, seperti tabel database, dan merekam metadata tentang perubahan tersebut. Anda dapat menggunakan CDC untuk berbagai tujuan, seperti mengaudit atau mereplikasi perubahan dalam sistem target untuk mempertahankan sinkronisasi.

rekayasa kekacauan

Sengaja memperkenalkan kegagalan atau peristiwa yang mengganggu untuk menguji ketahanan sistem. Anda dapat menggunakan AWS Fault Injection Service (AWS FIS) untuk melakukan eksperimen yang menekankan AWS beban kerja Anda dan mengevaluasi responsnya.

CI/CD

Lihat integrasi berkelanjutan dan pengiriman berkelanjutan.

klasifikasi

Proses kategorisasi yang membantu menghasilkan prediksi. Model ML untuk masalah klasifikasi memprediksi nilai diskrit. Nilai diskrit selalu berbeda satu sama lain. Misalnya, model mungkin perlu mengevaluasi apakah ada mobil dalam gambar atau tidak.

Enkripsi sisi klien

Enkripsi data secara lokal, sebelum target Layanan AWS menerimanya.

Pusat Keunggulan Cloud (CCoE)

Tim multi-disiplin yang mendorong upaya adopsi cloud di seluruh organisasi, termasuk mengembangkan praktik terbaik cloud, memobilisasi sumber daya, menetapkan jadwal migrasi, dan memimpin organisasi melalui transformasi skala besar. Untuk informasi selengkapnya, lihat posting CCo E di Blog Strategi AWS Cloud Perusahaan.

komputasi cloud

Teknologi cloud yang biasanya digunakan untuk penyimpanan data jarak jauh dan manajemen perangkat IoT. Cloud computing umumnya terhubung ke teknologi edge computing.

model operasi cloud

Dalam organisasi TI, model operasi yang digunakan untuk membangun, mematangkan, dan mengoptimalkan satu atau lebih lingkungan cloud. Untuk informasi selengkapnya, lihat Membangun Model Operasi Cloud Anda.

tahap adopsi cloud

Empat fase yang biasanya dilalui organisasi ketika mereka bermigrasi ke AWS Cloud:

- Proyek — Menjalankan beberapa proyek terkait cloud untuk bukti konsep dan tujuan pembelajaran
- Foundation — Melakukan investasi dasar untuk meningkatkan adopsi cloud Anda (misalnya, membuat landing zone, mendefinisikan CCo E, membuat model operasi)
- Migrasi — Migrasi aplikasi individual
- Re-invention — Mengoptimalkan produk dan layanan, dan berinovasi di cloud

Tahapan ini didefinisikan oleh Stephen Orban dalam posting blog The Journey Toward Cloud-First & the Stages of Adoption di blog Strategi Perusahaan. AWS Cloud Untuk informasi tentang bagaimana kaitannya dengan strategi AWS migrasi, lihat panduan kesiapan migrasi.

CMDB

Lihat database manajemen konfigurasi.

repositori kode

Lokasi di mana kode sumber dan aset lainnya, seperti dokumentasi, sampel, dan skrip, disimpan dan diperbarui melalui proses kontrol versi. Repositori cloud umum termasuk GitHub atau. Bitbucket Cloud Setiap versi kode disebut cabang. Dalam struktur layanan mikro, setiap repositori

dikhususkan untuk satu bagian fungsionalitas. Pipa CI/CD tunggal dapat menggunakan beberapa repositori.

cache dingin

Cache buffer yang kosong, tidak terisi dengan baik, atau berisi data basi atau tidak relevan. Ini mempengaruhi kinerja karena instance database harus membaca dari memori utama atau disk, yang lebih lambat daripada membaca dari cache buffer.

data dingin

Data yang jarang diakses dan biasanya historis. Saat menanyakan jenis data ini, kueri lambat biasanya dapat diterima. Memindahkan data ini ke tingkat penyimpanan atau kelas yang berkinerja lebih rendah dan lebih murah dapat mengurangi biaya.

visi komputer (CV)

Bidang AI yang menggunakan pembelajaran mesin untuk menganalisis dan mengekstrak informasi dari format visual seperti gambar dan video digital. Misalnya, Amazon SageMaker AI menyediakan algoritma pemrosesan gambar untuk CV.

konfigurasi drift

Untuk beban kerja, konfigurasi berubah dari status yang diharapkan. Ini dapat menyebabkan beban kerja menjadi tidak patuh, dan biasanya bertahap dan tidak disengaja.

database manajemen konfigurasi (CMDB)

Repositori yang menyimpan dan mengelola informasi tentang database dan lingkungan TI, termasuk komponen perangkat keras dan perangkat lunak dan konfigurasinya. Anda biasanya menggunakan data dari CMDB dalam penemuan portofolio dan tahap analisis migrasi.

paket kesesuaian

Kumpulan AWS Config aturan dan tindakan remediasi yang dapat Anda kumpulkan untuk menyesuaikan kepatuhan dan pemeriksaan keamanan Anda. Anda dapat menerapkan paket kesesuaian sebagai entitas tunggal di Akun AWS dan Wilayah, atau di seluruh organisasi, dengan menggunakan templat YAMM. Untuk informasi selengkapnya, lihat Paket kesesuaian dalam dokumentasi. AWS Config

integrasi berkelanjutan dan pengiriman berkelanjutan (CI/CD)

Proses mengotomatiskan sumber, membangun, menguji, pementasan, dan tahap produksi dari proses rilis perangkat lunak. CI/CD biasanya digambarkan sebagai pipa. CI/CD dapat membantu

Anda mengotomatiskan proses, meningkatkan produktivitas, meningkatkan kualitas kode, dan memberikan lebih cepat. Untuk informasi lebih lanjut, lihat Manfaat pengiriman berkelanjutan. CD juga dapat berarti penerapan berkelanjutan. Untuk informasi selengkapnya, lihat Continuous Delivery vs Continuous Deployment.

CV

Lihat visi komputer.

D

data saat istirahat

Data yang stasioner di jaringan Anda, seperti data yang ada di penyimpanan.

klasifikasi data

Proses untuk mengidentifikasi dan mengkategorikan data dalam jaringan Anda berdasarkan kekritisan dan sensitivitasnya. Ini adalah komponen penting dari setiap strategi manajemen risiko keamanan siber karena membantu Anda menentukan perlindungan dan kontrol retensi yang tepat untuk data. Klasifikasi data adalah komponen pilar keamanan dalam AWS Well-Architected Framework. Untuk informasi selengkapnya, lihat Klasifikasi data.

penyimpangan data

Variasi yang berarti antara data produksi dan data yang digunakan untuk melatih model ML, atau perubahan yang berarti dalam data input dari waktu ke waktu. Penyimpangan data dapat mengurangi kualitas, akurasi, dan keadilan keseluruhan dalam prediksi model ML.

data dalam transit

Data yang aktif bergerak melalui jaringan Anda, seperti antara sumber daya jaringan.

jala data

Kerangka arsitektur yang menyediakan kepemilikan data terdistribusi dan terdesentralisasi dengan manajemen dan tata kelola terpusat.

minimalisasi data

Prinsip pengumpulan dan pemrosesan hanya data yang sangat diperlukan. Mempraktikkan minimalisasi data di dalamnya AWS Cloud dapat mengurangi risiko privasi, biaya, dan jejak karbon analitik Anda.

perimeter data

Satu set pagar pembatas pencegahan di AWS lingkungan Anda yang membantu memastikan bahwa hanya identitas tepercaya yang mengakses sumber daya tepercaya dari jaringan yang diharapkan. Untuk informasi selengkapnya, lihat Membangun perimeter data pada AWS.

prapemrosesan data

Untuk mengubah data mentah menjadi format yang mudah diuraikan oleh model ML Anda. Preprocessing data dapat berarti menghapus kolom atau baris tertentu dan menangani nilai yang hilang, tidak konsisten, atau duplikat.

asal data

Proses melacak asal dan riwayat data sepanjang siklus hidupnya, seperti bagaimana data dihasilkan, ditransmisikan, dan disimpan.

subjek data

Individu yang datanya dikumpulkan dan diproses.

gudang data

Sistem manajemen data yang mendukung intelijen bisnis, seperti analitik. Gudang data biasanya berisi sejumlah besar data historis, dan biasanya digunakan untuk kueri dan analisis.

bahasa definisi database (DDL)

Pernyataan atau perintah untuk membuat atau memodifikasi struktur tabel dan objek dalam database.

bahasa manipulasi basis data (DHTML)

Pernyataan atau perintah untuk memodifikasi (memasukkan, memperbarui, dan menghapus) informasi dalam database.

DDL

Lihat bahasa definisi database.

ansambel yang dalam

Untuk menggabungkan beberapa model pembelajaran mendalam untuk prediksi. Anda dapat menggunakan ansambel dalam untuk mendapatkan prediksi yang lebih akurat atau untuk memperkirakan ketidakpastian dalam prediksi.

pembelajaran mendalam

Subbidang ML yang menggunakan beberapa lapisan jaringan saraf tiruan untuk mengidentifikasi pemetaan antara data input dan variabel target yang diinginkan.

defense-in-depth

Pendekatan keamanan informasi di mana serangkaian mekanisme dan kontrol keamanan dilapisi dengan cermat di seluruh jaringan komputer untuk melindungi kerahasiaan, integritas, dan ketersediaan jaringan dan data di dalamnya. Saat Anda mengadopsi strategi ini AWS, Anda menambahkan beberapa kontrol pada lapisan AWS Organizations struktur yang berbeda untuk membantu mengamankan sumber daya. Misalnya, defense-in-depth pendekatan mungkin menggabungkan otentikasi multi-faktor, segmentasi jaringan, dan enkripsi.

administrator yang didelegasikan

Di AWS Organizations, layanan yang kompatibel dapat mendaftarkan akun AWS anggota untuk mengelola akun organisasi dan mengelola izin untuk layanan tersebut. Akun ini disebut administrator yang didelegasikan untuk layanan itu. Untuk informasi selengkapnya dan daftar layanan yang kompatibel, lihat Layanan yang berfungsi dengan AWS Organizations AWS Organizations dokumentasi.

deployment

Proses pembuatan aplikasi, fitur baru, atau perbaikan kode tersedia di lingkungan target. Deployment melibatkan penerapan perubahan dalam basis kode dan kemudian membangun dan menjalankan basis kode itu di lingkungan aplikasi.

lingkungan pengembangan

Lihat lingkungan.

kontrol detektif

Kontrol keamanan yang dirancang untuk mendeteksi, mencatat, dan memperingatkan setelah suatu peristiwa terjadi. Kontrol ini adalah garis pertahanan kedua, memperingatkan Anda tentang peristiwa keamanan yang melewati kontrol pencegahan yang ada. Untuk informasi selengkapnya, lihat Kontrol Detektif dalam Menerapkan kontrol keamanan pada. AWS

pemetaan aliran nilai pengembangan (DVSM)

Sebuah proses yang digunakan untuk mengidentifikasi dan memprioritaskan kendala yang mempengaruhi kecepatan dan kualitas dalam siklus hidup pengembangan perangkat lunak. DVSM memperluas proses pemetaan aliran nilai yang awalnya dirancang untuk praktik

manufaktur ramping. Ini berfokus pada langkah-langkah dan tim yang diperlukan untuk menciptakan dan memindahkan nilai melalui proses pengembangan perangkat lunak.

kembar digital

Representasi virtual dari sistem dunia nyata, seperti bangunan, pabrik, peralatan industri, atau jalur produksi. Kembar digital mendukung pemeliharaan prediktif, pemantauan jarak jauh, dan optimalisasi produksi.

tabel dimensi

Dalam skema bintang, tabel yang lebih kecil yang berisi atribut data tentang data kuantitatif dalam tabel fakta. Atribut tabel dimensi biasanya bidang teks atau angka diskrit yang berperilaku seperti teks. Atribut ini biasanya digunakan untuk pembatasan kueri, pemfilteran, dan pelabelan set hasil.

musibah

Peristiwa yang mencegah beban kerja atau sistem memenuhi tujuan bisnisnya di lokasi utama yang digunakan. Peristiwa ini dapat berupa bencana alam, kegagalan teknis, atau akibat dari tindakan manusia, seperti kesalahan konfigurasi yang tidak disengaja atau serangan malware.

pemulihan bencana (DR)

Strategi dan proses yang Anda gunakan untuk meminimalkan downtime dan kehilangan data yang disebabkan oleh bencana. Untuk informasi selengkapnya, lihat Disaster Recovery of Workloads on AWS: Recovery in the Cloud in the AWS Well-Architected Framework.

DML~

Lihat bahasa manipulasi basis data.

desain berbasis domain

Pendekatan untuk mengembangkan sistem perangkat lunak yang kompleks dengan menghubungkan komponennya ke domain yang berkembang, atau tujuan bisnis inti, yang dilayani oleh setiap komponen. Konsep ini diperkenalkan oleh Eric Evans dalam bukunya, Domain-Driven Design: Tackling Complexity in the Heart of Software (Boston: Addison-Wesley Professional, 2003). Untuk informasi tentang cara menggunakan desain berbasis domain dengan pola gambar pencekik, lihat Memodernisasi layanan web Microsoft ASP.NET (ASMX) lama secara bertahap menggunakan container dan Amazon API Gateway.

DR

Lihat pemulihan bencana.

deteksi drift

Melacak penyimpangan dari konfigurasi dasar. Misalnya, Anda dapat menggunakan AWS CloudFormation untuk mendeteksi penyimpangan dalam sumber daya sistem, atau Anda dapat menggunakannya AWS Control Tower untuk mendeteksi perubahan di landing zone yang mungkin memengaruhi kepatuhan terhadap persyaratan tata kelola.

DVSM

Lihat pemetaan aliran nilai pengembangan.

E

EDA

Lihat analisis data eksplorasi.

EDI

Lihat pertukaran data elektronik.

komputasi tepi

Teknologi yang meningkatkan daya komputasi untuk perangkat pintar di tepi jaringan IoT. Jika dibandingkan dengan komputasi awan, komputasi tepi dapat mengurangi latensi komunikasi dan meningkatkan waktu respons.

pertukaran data elektronik (EDI)

Pertukaran otomatis dokumen bisnis antar organisasi. Untuk informasi selengkapnya, lihat Apa itu Pertukaran Data Elektronik.

enkripsi

Proses komputasi yang mengubah data plaintext, yang dapat dibaca manusia, menjadi ciphertext.

kunci enkripsi

String kriptografi dari bit acak yang dihasilkan oleh algoritma enkripsi. Panjang kunci dapat bervariasi, dan setiap kunci dirancang agar tidak dapat diprediksi dan unik.

endianness

Urutan byte disimpan dalam memori komputer. Sistem big-endian menyimpan byte paling signifikan terlebih dahulu. Sistem little-endian menyimpan byte paling tidak signifikan terlebih dahulu.

titik akhir

Lihat titik akhir layanan.

layanan endpoint

Layanan yang dapat Anda host di cloud pribadi virtual (VPC) untuk dibagikan dengan pengguna lain. Anda dapat membuat layanan endpoint dengan AWS PrivateLink dan memberikan izin kepada prinsipal lain Akun AWS atau ke AWS Identity and Access Management (IAM). Akun atau prinsipal ini dapat terhubung ke layanan endpoint Anda secara pribadi dengan membuat titik akhir VPC antarmuka. Untuk informasi selengkapnya, lihat Membuat layanan titik akhir di dokumentasi Amazon Virtual Private Cloud (Amazon VPC).

perencanaan sumber daya perusahaan (ERP)

Sistem yang mengotomatiskan dan mengelola proses bisnis utama (seperti akuntansi, MES, dan manajemen proyek) untuk suatu perusahaan.

enkripsi amplop

Proses mengenkripsi kunci enkripsi dengan kunci enkripsi lain. Untuk informasi selengkapnya, lihat Enkripsi amplop dalam dokumentasi AWS Key Management Service (AWS KMS).

lingkungan

Sebuah contoh dari aplikasi yang sedang berjalan. Berikut ini adalah jenis lingkungan yang umum dalam komputasi awan:

- Development Environment — Sebuah contoh dari aplikasi yang berjalan yang hanya tersedia untuk tim inti yang bertanggung jawab untuk memelihara aplikasi. Lingkungan pengembangan digunakan untuk menguji perubahan sebelum mempromosikannya ke lingkungan atas. Jenis lingkungan ini kadang-kadang disebut sebagai lingkungan pengujian.
- lingkungan yang lebih rendah — Semua lingkungan pengembangan untuk aplikasi, seperti yang digunakan untuk build awal dan pengujian.
- lingkungan produksi — Sebuah contoh dari aplikasi yang berjalan yang pengguna akhir dapat mengakses. Dalam sebuah CI/CD pipeline, lingkungan produksi adalah lingkungan penyebaran terakhir.

- lingkungan atas — Semua lingkungan yang dapat diakses oleh pengguna selain tim pengembangan inti. Ini dapat mencakup lingkungan produksi, lingkungan praproduksi, dan lingkungan untuk pengujian penerimaan pengguna.

epik

Dalam metodologi tangkas, kategori fungsional yang membantu mengatur dan memprioritaskan pekerjaan Anda. Epik memberikan deskripsi tingkat tinggi tentang persyaratan dan tugas implementasi. Misalnya, epos keamanan AWS CAF mencakup manajemen identitas dan akses, kontrol detektif, keamanan infrastruktur, perlindungan data, dan respons insiden. Untuk informasi selengkapnya tentang epos dalam strategi AWS migrasi, lihat panduan implementasi program.

ERP

Lihat perencanaan sumber daya perusahaan.

analisis data eksplorasi (EDA)

Proses menganalisis dataset untuk memahami karakteristik utamanya. Anda mengumpulkan atau mengumpulkan data dan kemudian melakukan penyelidikan awal untuk menemukan pola, mendeteksi anomali, dan memeriksa asumsi. EDA dilakukan dengan menghitung statistik ringkasan dan membuat visualisasi data.

F

tabel fakta

Tabel tengah dalam skema bintang. Ini menyimpan data kuantitatif tentang operasi bisnis. Biasanya, tabel fakta berisi dua jenis kolom: kolom yang berisi ukuran dan yang berisi kunci asing ke tabel dimensi.

gagal cepat

Filosofi yang menggunakan pengujian yang sering dan bertahap untuk mengurangi siklus hidup pengembangan. Ini adalah bagian penting dari pendekatan tangkas.

batas isolasi kesalahan

Dalam AWS Cloud, batas seperti Availability Zone, Wilayah AWS, control plane, atau data plane yang membatasi efek kegagalan dan membantu meningkatkan ketahanan beban kerja. Untuk informasi selengkapnya, lihat Batas Isolasi AWS Kesalahan.

cabang fitur

Lihat cabang.

fitur

Data input yang Anda gunakan untuk membuat prediksi. Misalnya, dalam konteks manufaktur, fitur bisa berupa gambar yang diambil secara berkala dari lini manufaktur.

pentingnya fitur

Seberapa signifikan fitur untuk prediksi model. Ini biasanya dinyatakan sebagai skor numerik yang dapat dihitung melalui berbagai teknik, seperti Shapley Additive Explanations (SHAP) dan gradien terintegrasi. Untuk informasi lebih lanjut, lihat Interpretabilitas model pembelajaran mesin dengan. AWS

transformasi fitur

Untuk mengoptimalkan data untuk proses ML, termasuk memperkaya data dengan sumber tambahan, menskalakan nilai, atau mengekstrak beberapa set informasi dari satu bidang data. Hal ini memungkinkan model ML untuk mendapatkan keuntungan dari data. Misalnya, jika Anda memecah tanggal “2021-05-27 00:15:37" menjadi “2021", “Mei”, “Kamis”, dan “15", Anda dapat membantu algoritme pembelajaran mempelajari pola bernuansa yang terkait dengan komponen data yang berbeda.

beberapa tembakan mendorong

Menyediakan LLM dengan sejumlah kecil contoh yang menunjukkan tugas dan output yang diinginkan sebelum memintanya untuk melakukan tugas serupa. Teknik ini adalah aplikasi pembelajaran dalam konteks, di mana model belajar dari contoh (bidikan) yang tertanam dalam petunjuk. Beberapa bidikan dapat efektif untuk tugas-tugas yang memerlukan pemformatan, penalaran, atau pengetahuan domain tertentu. Lihat juga zero-shot prompting.

FGAC

Lihat kontrol akses berbutir halus.

kontrol akses berbutir halus (FGAC)

Penggunaan beberapa kondisi untuk mengizinkan atau menolak permintaan akses.

migrasi flash-cut

Metode migrasi database yang menggunakan replikasi data berkelanjutan melalui pengambilan data perubahan untuk memigrasikan data dalam waktu sesingkat mungkin, alih-alih

menggunakan pendekatan bertahap. Tujuannya adalah untuk menjaga downtime seminimal mungkin.

FM

Lihat model pondasi.

model pondasi (FM)

Jaringan saraf pembelajaran mendalam yang besar yang telah melatih kumpulan data besar-besaran data umum dan tidak berlabel. FMs mampu melakukan berbagai tugas umum, seperti memahami bahasa, menghasilkan teks dan gambar, dan berbicara dalam bahasa alami. Untuk informasi selengkapnya, lihat Apa itu Model Foundation.

G

AI generatif

Subset model AI yang telah dilatih pada sejumlah besar data dan yang dapat menggunakan prompt teks sederhana untuk membuat konten dan artefak baru, seperti gambar, video, teks, dan audio. Untuk informasi lebih lanjut, lihat Apa itu AI Generatif.

pemblokiran geografis

Lihat pembatasan geografis.

pembatasan geografis (pemblokiran geografis)

Di Amazon CloudFront, opsi untuk mencegah pengguna di negara tertentu mengakses distribusi konten. Anda dapat menggunakan daftar izinkan atau daftar blokir untuk menentukan negara yang disetujui dan dilarang. Untuk informasi selengkapnya, lihat Membatasi distribusi geografis konten Anda dalam dokumentasi. CloudFront

Alur kerja Gitflow

Pendekatan di mana lingkungan bawah dan atas menggunakan cabang yang berbeda dalam repositori kode sumber. Alur kerja Gitflow dianggap warisan, dan alur kerja berbasis batang adalah pendekatan modern yang lebih disukai.

gambar emas

Sebuah snapshot dari sistem atau perangkat lunak yang digunakan sebagai template untuk menyebarkan instance baru dari sistem atau perangkat lunak itu. Misalnya, di bidang manufaktur,

gambar emas dapat digunakan untuk menyediakan perangkat lunak pada beberapa perangkat dan membantu meningkatkan kecepatan, skalabilitas, dan produktivitas dalam operasi manufaktur perangkat.

strategi greenfield

Tidak adanya infrastruktur yang ada di lingkungan baru. Saat mengadopsi strategi greenfield untuk arsitektur sistem, Anda dapat memilih semua teknologi baru tanpa batasan kompatibilitas dengan infrastruktur yang ada, juga dikenal sebagai brownfield. Jika Anda memperluas infrastruktur yang ada, Anda dapat memadukan strategi brownfield dan greenfield.

pagar pembatas

Aturan tingkat tinggi yang membantu mengatur sumber daya, kebijakan, dan kepatuhan di seluruh unit organisasi ()OUs. Pagar pembatas preventif menegakkan kebijakan untuk memastikan keselarasan dengan standar kepatuhan. Mereka diimplementasikan dengan menggunakan kebijakan kontrol layanan dan batas izin IAM. Detective guardrails mendeteksi pelanggaran kebijakan dan masalah kepatuhan, dan menghasilkan peringatan untuk remediasi. Mereka diimplementasikan dengan menggunakan AWS Config, AWS Security Hub CSPM, Amazon GuardDuty AWS Trusted Advisor, Amazon Inspector, dan pemeriksaan khusus AWS Lambda .

H

HA

Lihat ketersediaan tinggi.

migrasi database heterogen

Memigrasi database sumber Anda ke database target yang menggunakan mesin database yang berbeda (misalnya, Oracle ke Amazon Aurora). Migrasi heterogen biasanya merupakan bagian dari upaya arsitektur ulang, dan mengubah skema dapat menjadi tugas yang kompleks. AWS menyediakan AWS SCT yang membantu dengan konversi skema.

ketersediaan tinggi (HA)

Kemampuan beban kerja untuk beroperasi terus menerus, tanpa intervensi, jika terjadi tantangan atau bencana. Sistem HA dirancang untuk gagal secara otomatis, secara konsisten memberikan kinerja berkualitas tinggi, dan menangani beban dan kegagalan yang berbeda dengan dampak kinerja minimal.

modernisasi sejarawan

Pendekatan yang digunakan untuk memodernisasi dan meningkatkan sistem teknologi operasional (OT) untuk melayani kebutuhan industri manufaktur dengan lebih baik. Sejarawan adalah jenis database yang digunakan untuk mengumpulkan dan menyimpan data dari berbagai sumber di pabrik.

data penahanan

Sebagian dari data historis berlabel yang ditahan dari kumpulan data yang digunakan untuk melatih model pembelajaran mesin. Anda dapat menggunakan data penahanan untuk mengevaluasi kinerja model dengan membandingkan prediksi model dengan data penahanan.

migrasi database homogen

Memigrasi database sumber Anda ke database target yang berbagi mesin database yang sama (misalnya, Microsoft SQL Server ke Amazon RDS for SQL Server). Migrasi homogen biasanya merupakan bagian dari upaya rehosting atau replatforming. Anda dapat menggunakan utilitas database asli untuk memigrasi skema.

data panas

Data yang sering diakses, seperti data real-time atau data translasi terbaru. Data ini biasanya memerlukan tingkat atau kelas penyimpanan berkinerja tinggi untuk memberikan respons kueri yang cepat.

perbaikan terbaru

Perbaikan mendesak untuk masalah kritis dalam lingkungan produksi. Karena urgensinya, perbaikan terbaru biasanya dibuat di luar alur kerja DevOps rilis biasa.

periode hypercare

Segera setelah cutover, periode waktu ketika tim migrasi mengelola dan memantau aplikasi yang dimigrasi di cloud untuk mengatasi masalah apa pun. Biasanya, periode ini panjangnya 1-4 hari. Pada akhir periode hypercare, tim migrasi biasanya mentransfer tanggung jawab untuk aplikasi ke tim operasi cloud.

I

IAc

Lihat infrastruktur sebagai kode.

kebijakan berbasis identitas

Kebijakan yang dilampirkan pada satu atau beberapa prinsip IAM yang mendefinisikan izin mereka dalam lingkungan. AWS Cloud

aplikasi idle

Aplikasi yang memiliki penggunaan CPU dan memori rata-rata antara 5 dan 20 persen selama periode 90 hari. Dalam proyek migrasi, adalah umum untuk menghentikan aplikasi ini atau mempertahankannya di tempat.

IIoT

Lihat Internet of Things industri.

infrastruktur yang tidak dapat diubah

Model yang menyebarkan infrastruktur baru untuk beban kerja produksi alih-alih memperbarui, menambal, atau memodifikasi infrastruktur yang ada. Infrastruktur yang tidak dapat diubah secara inheren lebih konsisten, andal, dan dapat diprediksi daripada infrastruktur yang dapat berubah. Untuk informasi selengkapnya, lihat praktik terbaik Deploy using immutable infrastructure di AWS Well-Architected Framework.

masuk (masuknya) VPC

Dalam arsitektur AWS multi-akun, VPC yang menerima, memeriksa, dan merutekan koneksi jaringan dari luar aplikasi. Arsitektur Referensi AWS Keamanan merekomendasikan pengaturan akun Jaringan Anda dengan inbound, outbound, dan inspeksi VPCs untuk melindungi antarmuka dua arah antara aplikasi Anda dan internet yang lebih luas.

migrasi inkremental

Strategi cutover di mana Anda memigrasikan aplikasi Anda dalam bagian-bagian kecil alih-alih melakukan satu cutover penuh. Misalnya, Anda mungkin hanya memindahkan beberapa layanan mikro atau pengguna ke sistem baru pada awalnya. Setelah Anda memverifikasi bahwa semuanya berfungsi dengan baik, Anda dapat secara bertahap memindahkan layanan mikro atau pengguna tambahan hingga Anda dapat menonaktifkan sistem lama Anda. Strategi ini mengurangi risiko yang terkait dengan migrasi besar.

Industri 4.0

Sebuah istilah yang diperkenalkan oleh Klaus Schwab pada tahun 2016 untuk merujuk pada modernisasi proses manufaktur melalui kemajuan dalam konektivitas, data real-time, otomatisasi, analitik, dan AI/ML.

infrastruktur

Semua sumber daya dan aset yang terkandung dalam lingkungan aplikasi.

infrastruktur sebagai kode (IAc)

Proses penyediaan dan pengelolaan infrastruktur aplikasi melalui satu set file konfigurasi. IAc dirancang untuk membantu Anda memusatkan manajemen infrastruktur, menstandarisasi sumber daya, dan menskalakan dengan cepat sehingga lingkungan baru dapat diulang, andal, dan konsisten.

Internet of Things industri (IIoT)

Penggunaan sensor dan perangkat yang terhubung ke internet di sektor industri, seperti manufaktur, energi, otomotif, perawatan kesehatan, ilmu kehidupan, dan pertanian. Untuk informasi lebih lanjut, lihat Membangun strategi transformasi digital Internet of Things (IIoT) industri.

inspeksi VPC

Dalam arsitektur AWS multi-akun, VPC terpusat yang mengelola inspeksi lalu lintas jaringan antara VPCs (dalam yang sama atau berbeda Wilayah AWS), internet, dan jaringan lokal. Arsitektur Referensi AWS Keamanan merekomendasikan pengaturan akun Jaringan Anda dengan inbound, outbound, dan inspeksi VPCs untuk melindungi antarmuka dua arah antara aplikasi Anda dan internet yang lebih luas.

Internet of Things (IoT)

Jaringan objek fisik yang terhubung dengan sensor atau prosesor tertanam yang berkomunikasi dengan perangkat dan sistem lain melalui internet atau melalui jaringan komunikasi lokal. Untuk informasi selengkapnya, lihat Apa itu IoT?

interpretasi

Karakteristik model pembelajaran mesin yang menggambarkan sejauh mana manusia dapat memahami bagaimana prediksi model bergantung pada inputnya. Untuk informasi lebih lanjut, lihat Interpretabilitas model pembelajaran mesin dengan. AWS

IoT

Lihat Internet of Things.

Perpustakaan informasi TI (ITIL)

Serangkaian praktik terbaik untuk memberikan layanan TI dan menyelaraskan layanan ini dengan persyaratan bisnis. ITIL menyediakan dasar untuk ITSM.

Manajemen layanan TI (ITSM)

Kegiatan yang terkait dengan merancang, menerapkan, mengelola, dan mendukung layanan TI untuk suatu organisasi. Untuk informasi tentang mengintegrasikan operasi cloud dengan alat ITSM, lihat panduan integrasi operasi.

ITIL

Lihat perpustakaan informasi TI.

ITSM

Lihat manajemen layanan TI.

L

kontrol akses berbasis label (LBAC)

Implementasi kontrol akses wajib (MAC) di mana pengguna dan data itu sendiri masing-masing secara eksplisit diberi nilai label keamanan. Persimpangan antara label keamanan pengguna dan label keamanan data menentukan baris dan kolom mana yang dapat dilihat oleh pengguna.

landing zone

Landing zone adalah AWS lingkungan multi-akun yang dirancang dengan baik yang dapat diskalakan dan aman. Ini adalah titik awal dari mana organisasi Anda dapat dengan cepat meluncurkan dan menyebarkan beban kerja dan aplikasi dengan percaya diri dalam lingkungan keamanan dan infrastruktur mereka. Untuk informasi selengkapnya tentang zona pendaratan, lihat Menyiapkan lingkungan multi-akun AWS yang aman dan dapat diskalakan.

model bahasa besar (LLM)

Model AI pembelajaran mendalam yang dilatih sebelumnya pada sejumlah besar data. LLM dapat melakukan beberapa tugas, seperti menjawab pertanyaan, meringkas dokumen, menerjemahkan teks ke dalam bahasa lain, dan menyelesaikan kalimat. Untuk informasi lebih lanjut, lihat Apa itu LLMs.

migrasi besar

Migrasi 300 atau lebih server.

LBAC

Lihat kontrol akses berbasis label.

hak istimewa paling sedikit

Praktik keamanan terbaik untuk memberikan izin minimum yang diperlukan untuk melakukan tugas. Untuk informasi selengkapnya, lihat Menerapkan izin hak istimewa terkecil dalam dokumentasi IAM.

angkat dan geser

Lihat 7 Rs.

sistem endian kecil

Sebuah sistem yang menyimpan byte paling tidak signifikan terlebih dahulu. Lihat juga endianness.

LLM

Lihat model bahasa besar.

lingkungan yang lebih rendah

Lihat lingkungan.

M

pembelajaran mesin (ML)

Jenis kecerdasan buatan yang menggunakan algoritma dan teknik untuk pengenalan pola dan pembelajaran. ML menganalisis dan belajar dari data yang direkam, seperti data Internet of Things (IoT), untuk menghasilkan model statistik berdasarkan pola. Untuk informasi selengkapnya, lihat Machine Learning.

cabang utama

Lihat cabang.

malware

Perangkat lunak yang dirancang untuk membahayakan keamanan atau privasi komputer. Malware dapat mengganggu sistem komputer, membocorkan informasi sensitif, atau mendapatkan akses yang tidak sah. Contoh malware termasuk virus, worm, ransomware, Trojan horse, spyware, dan keyloggers.

layanan terkelola

Layanan AWS yang AWS mengoperasikan lapisan infrastruktur, sistem operasi, dan platform, dan Anda mengakses titik akhir untuk menyimpan dan mengambil data. Amazon Simple Storage Service (Amazon S3) dan Amazon DynamoDB adalah contoh layanan terkelola. Ini juga dikenal sebagai layanan abstrak.

sistem eksekusi manufaktur (MES)

Sistem perangkat lunak untuk melacak, memantau, mendokumentasikan, dan mengendalikan proses produksi yang mengubah bahan baku menjadi produk jadi di lantai toko.

PETA

Lihat Program Percepatan Migrasi.

mekanisme

Proses lengkap di mana Anda membuat alat, mendorong adopsi alat, dan kemudian memeriksa hasilnya untuk melakukan penyesuaian. Mekanisme adalah siklus yang memperkuat dan meningkatkan dirinya sendiri saat beroperasi. Untuk informasi lebih lanjut, lihat Membangun mekanisme di AWS Well-Architected Framework.

akun anggota

Semua Akun AWS selain akun manajemen yang merupakan bagian dari organisasi di AWS Organizations. Akun dapat menjadi anggota dari hanya satu organisasi pada suatu waktu.

MES

Lihat sistem eksekusi manufaktur.

Transportasi Telemetri Antrian Pesan (MQTT)

Protokol komunikasi ringan machine-to-machine (M2M), berdasarkan pola terbitkan/berlangganan, untuk perangkat IoT yang dibatasi sumber daya.

layanan mikro

Layanan kecil dan independen yang berkomunikasi dengan jelas APIs dan biasanya dimiliki oleh tim kecil yang mandiri. Misalnya, sistem asuransi mungkin mencakup layanan mikro yang memetakan kemampuan bisnis, seperti penjualan atau pemasaran, atau subdomain, seperti pembelian, klaim, atau analitik. Manfaat layanan mikro termasuk kelincahan, penskalaan yang fleksibel, penyebaran yang mudah, kode yang dapat digunakan kembali, dan ketahanan. Untuk informasi selengkapnya, lihat Mengintegrasikan layanan mikro dengan menggunakan layanan tanpa AWS server.

arsitektur microservices

Pendekatan untuk membangun aplikasi dengan komponen independen yang menjalankan setiap proses aplikasi sebagai layanan mikro. Layanan mikro ini berkomunikasi melalui antarmuka yang terdefinisi dengan baik dengan menggunakan ringan. APIs Setiap layanan mikro dalam arsitektur ini dapat diperbarui, digunakan, dan diskalakan untuk memenuhi permintaan fungsi tertentu dari suatu aplikasi. Untuk informasi selengkapnya, lihat Menerapkan layanan mikro di AWS.

Program Percepatan Migrasi (MAP)

AWS Program yang menyediakan dukungan konsultasi, pelatihan, dan layanan untuk membantu organisasi membangun fondasi operasional yang kuat untuk pindah ke cloud, dan untuk membantu mengimbangi biaya awal migrasi. MAP mencakup metodologi migrasi untuk mengeksekusi migrasi lama dengan cara metodis dan seperangkat alat untuk mengotomatisasi dan mempercepat skenario migrasi umum.

migrasi dalam skala

Proses memindahkan sebagian besar portofolio aplikasi ke cloud dalam gelombang, dengan lebih banyak aplikasi bergerak pada tingkat yang lebih cepat di setiap gelombang. Fase ini menggunakan praktik dan pelajaran terbaik dari fase sebelumnya untuk mengimplementasikan pabrik migrasi tim, alat, dan proses untuk merampingkan migrasi beban kerja melalui otomatisasi dan pengiriman tangkas. Ini adalah fase ketiga dari strategi AWS migrasi.

pabrik migrasi

Tim lintas fungsi yang merampingkan migrasi beban kerja melalui pendekatan otomatis dan gesit. Tim pabrik migrasi biasanya mencakup operasi, analis dan pemilik bisnis, insinyur migrasi, pengembang, dan DevOps profesional yang bekerja di sprint. Antara 20 dan 50 persen portofolio aplikasi perusahaan terdiri dari pola berulang yang dapat dioptimalkan dengan pendekatan pabrik. Untuk informasi selengkapnya, lihat diskusi tentang pabrik migrasi dan panduan Pabrik Migrasi Cloud di kumpulan konten ini.

metadata migrasi

Informasi tentang aplikasi dan server yang diperlukan untuk menyelesaikan migrasi. Setiap pola migrasi memerlukan satu set metadata migrasi yang berbeda. Contoh metadata migrasi termasuk subnet target, grup keamanan, dan akun. AWS

pola migrasi

Tugas migrasi berulang yang merinci strategi migrasi, tujuan migrasi, dan aplikasi atau layanan migrasi yang digunakan. Contoh: Rehost migrasi ke Amazon EC2 AWS dengan Layanan Migrasi Aplikasi.

Penilaian Portofolio Migrasi (MPA)

Alat online yang menyediakan informasi untuk memvalidasi kasus bisnis untuk bermigrasi ke. AWS Cloud MPA menyediakan penilaian portofolio terperinci (ukuran kanan server, harga, perbandingan TCO, analisis biaya migrasi) serta perencanaan migrasi (analisis data aplikasi dan pengumpulan data, pengelompokan aplikasi, prioritas migrasi, dan perencanaan gelombang). Alat MPA (memerlukan login) tersedia gratis untuk semua AWS konsultan dan konsultan APN Partner.

Penilaian Kesiapan Migrasi (MRA)

Proses mendapatkan wawasan tentang status kesiapan cloud organisasi, mengidentifikasi kekuatan dan kelemahan, dan membangun rencana aksi untuk menutup kesenjangan yang diidentifikasi, menggunakan CAF. AWS Untuk informasi selengkapnya, lihat panduan kesiapan migrasi. MRA adalah tahap pertama dari strategi AWS migrasi.

strategi migrasi

Pendekatan yang digunakan untuk memigrasikan beban kerja ke file. AWS Cloud Untuk informasi lebih lanjut, lihat entri 7 Rs di glosarium ini dan lihat Memobilisasi organisasi Anda untuk mempercepat migrasi skala besar.

ML

Lihat pembelajaran mesin.

modernisasi

Mengubah aplikasi usang (warisan atau monolitik) dan infrastrukturnya menjadi sistem yang gesit, elastis, dan sangat tersedia di cloud untuk mengurangi biaya, mendapatkan efisiensi, dan memanfaatkan inovasi. Untuk informasi selengkapnya, lihat Strategi untuk memodernisasi aplikasi di. AWS Cloud

penilaian kesiapan modernisasi

Evaluasi yang membantu menentukan kesiapan modernisasi aplikasi organisasi; mengidentifikasi manfaat, risiko, dan dependensi; dan menentukan seberapa baik organisasi dapat mendukung keadaan masa depan aplikasi tersebut. Hasil penilaian adalah cetak biru arsitektur target, peta jalan yang merinci fase pengembangan dan tonggak untuk proses modernisasi, dan rencana aksi untuk mengatasi kesenjangan yang diidentifikasi. Untuk informasi lebih lanjut, lihat Mengevaluasi kesiapan modernisasi untuk aplikasi di. AWS Cloud

aplikasi monolitik (monolit)

Aplikasi yang berjalan sebagai layanan tunggal dengan proses yang digabungkan secara ketat. Aplikasi monolitik memiliki beberapa kelemahan. Jika satu fitur aplikasi mengalami lonjakan permintaan, seluruh arsitektur harus diskalakan. Menambahkan atau meningkatkan fitur aplikasi monolitik juga menjadi lebih kompleks ketika basis kode tumbuh. Untuk mengatasi masalah ini, Anda dapat menggunakan arsitektur microservices. Untuk informasi lebih lanjut, lihat Mengurai monolit menjadi layanan mikro.

MPA

Lihat Penilaian Portofolio Migrasi.

MQTT

Lihat Transportasi Telemetri Antrian Pesan.

klasifikasi multiclass

Sebuah proses yang membantu menghasilkan prediksi untuk beberapa kelas (memprediksi satu dari lebih dari dua hasil). Misalnya, model ML mungkin bertanya “Apakah produk ini buku, mobil, atau telepon?” atau “Kategori produk mana yang paling menarik bagi pelanggan ini?”

infrastruktur yang bisa berubah

Model yang memperbarui dan memodifikasi infrastruktur yang ada untuk beban kerja produksi. Untuk meningkatkan konsistensi, keandalan, dan prediktabilitas, AWS Well-Architected Framework merekomendasikan penggunaan infrastruktur yang tidak dapat diubah sebagai praktik terbaik.

O

OAC

Lihat kontrol akses asal.

OAI

Lihat identitas akses asal.

OCM

Lihat manajemen perubahan organisasi.

migrasi offline

Metode migrasi di mana beban kerja sumber diturunkan selama proses migrasi. Metode ini melibatkan waktu henti yang diperpanjang dan biasanya digunakan untuk beban kerja kecil dan tidak kritis.

OI

Lihat integrasi operasi.

OLA

Lihat perjanjian tingkat operasional.

migrasi online

Metode migrasi di mana beban kerja sumber disalin ke sistem target tanpa diambil offline. Aplikasi yang terhubung ke beban kerja dapat terus berfungsi selama migrasi. Metode ini melibatkan waktu henti nol hingga minimal dan biasanya digunakan untuk beban kerja produksi yang kritis.

OPC-UA

Lihat Komunikasi Proses Terbuka - Arsitektur Terpadu.

Komunikasi Proses Terbuka - Arsitektur Terpadu (OPC-UA)

Protokol komunikasi machine-to-machine (M2M) untuk otomasi industri. OPC-UA menyediakan standar interoperabilitas dengan enkripsi data, otentikasi, dan skema otorisasi.

perjanjian tingkat operasional (OLA)

Perjanjian yang menjelaskan apa yang dijanjikan kelompok TI fungsional untuk diberikan satu sama lain, untuk mendukung perjanjian tingkat layanan (SLA).

Tinjauan Kesiapan Operasional (ORR)

Daftar pertanyaan dan praktik terbaik terkait yang membantu Anda memahami, mengevaluasi, mencegah, atau mengurangi ruang lingkup insiden dan kemungkinan kegagalan. Untuk informasi lebih lanjut, lihat Ulasan Kesiapan Operasional (ORR) dalam Kerangka Kerja Well-Architected AWS .

teknologi operasional (OT)

Sistem perangkat keras dan perangkat lunak yang bekerja dengan lingkungan fisik untuk mengendalikan operasi industri, peralatan, dan infrastruktur. Di bidang manufaktur, integrasi sistem OT dan teknologi informasi (TI) adalah fokus utama untuk transformasi Industri 4.0.

integrasi operasi (OI)

Proses modernisasi operasi di cloud, yang melibatkan perencanaan kesiapan, otomatisasi, dan integrasi. Untuk informasi selengkapnya, lihat panduan integrasi operasi.

jejak organisasi

Jejak yang dibuat oleh AWS CloudTrail itu mencatat semua peristiwa untuk semua Akun AWS dalam organisasi di AWS Organizations. Jejak ini dibuat di setiap Akun AWS bagian organisasi dan melacak aktivitas di setiap akun. Untuk informasi selengkapnya, lihat Membuat jejak untuk organisasi dalam CloudTrail dokumentasi.

manajemen perubahan organisasi (OCM)

Kerangka kerja untuk mengelola transformasi bisnis utama yang mengganggu dari perspektif orang, budaya, dan kepemimpinan. OCM membantu organisasi mempersiapkan, dan transisi ke, sistem dan strategi baru dengan mempercepat adopsi perubahan, mengatasi masalah transisi, dan mendorong perubahan budaya dan organisasi. Dalam strategi AWS migrasi, kerangka kerja ini disebut percepatan orang, karena kecepatan perubahan yang diperlukan dalam proyek adopsi cloud. Untuk informasi lebih lanjut, lihat panduan OCM.

kontrol akses asal (OAC)

Di CloudFront, opsi yang disempurnakan untuk membatasi akses untuk mengamankan konten Amazon Simple Storage Service (Amazon S3) Anda. OAC mendukung semua bucket S3 di semua Wilayah AWS, enkripsi sisi server dengan AWS KMS (SSE-KMS), dan dinamis dan permintaan ke bucket S3. `PUT` `DELETE`

identitas akses asal (OAI)

Di CloudFront, opsi untuk membatasi akses untuk mengamankan konten Amazon S3 Anda. Saat Anda menggunakan OAI, CloudFront buat prinsipal yang dapat diautentikasi oleh Amazon S3. Prinsipal yang diautentikasi dapat mengakses konten dalam bucket S3 hanya melalui distribusi tertentu. CloudFront Lihat juga OAC, yang menyediakan kontrol akses yang lebih terperinci dan ditingkatkan.

ORR

Lihat tinjauan kesiapan operasional.

OT

Lihat teknologi operasional.

keluar (jalan keluar) VPC

Dalam arsitektur AWS multi-akun, VPC yang menangani koneksi jaringan yang dimulai dari dalam aplikasi. Arsitektur Referensi AWS Keamanan merekomendasikan pengaturan akun Jaringan Anda dengan inbound, outbound, dan inspeksi VPCs untuk melindungi antarmuka dua arah antara aplikasi Anda dan internet yang lebih luas.

P

batas izin

Kebijakan manajemen IAM yang dilampirkan pada prinsipal IAM untuk menetapkan izin maksimum yang dapat dimiliki pengguna atau peran. Untuk informasi selengkapnya, lihat Batas izin dalam dokumentasi IAM.

Informasi Identifikasi Pribadi (PII)

Informasi yang, jika dilihat secara langsung atau dipasangkan dengan data terkait lainnya, dapat digunakan untuk menyimpulkan identitas individu secara wajar. Contoh PII termasuk nama, alamat, dan informasi kontak.

PII

Lihat informasi yang dapat diidentifikasi secara pribadi.

buku pedoman

Serangkaian langkah yang telah ditentukan sebelumnya yang menangkap pekerjaan yang terkait dengan migrasi, seperti mengirimkan fungsi operasi inti di cloud. Buku pedoman dapat berupa skrip, runbook otomatis, atau ringkasan proses atau langkah-langkah yang diperlukan untuk mengoperasikan lingkungan modern Anda.

PLC

Lihat pengontrol logika yang dapat diprogram.

PLM

Lihat manajemen siklus hidup produk.

kebijakan

Objek yang dapat menentukan izin (lihat kebijakan berbasis identitas), menentukan kondisi akses (lihat kebijakanberbasis sumber daya), atau menentukan izin maksimum untuk semua akun dalam organisasi di (lihat kebijakan kontrol layanan). AWS Organizations

persistensi poliglot

Secara independen memilih teknologi penyimpanan data microservice berdasarkan pola akses data dan persyaratan lainnya. Jika layanan mikro Anda memiliki teknologi penyimpanan data yang sama, mereka dapat menghadapi tantangan implementasi atau mengalami kinerja yang buruk. Layanan mikro lebih mudah diimplementasikan dan mencapai kinerja dan skalabilitas yang lebih baik jika mereka menggunakan penyimpanan data yang paling sesuai dengan kebutuhan mereka.

penilaian portofolio

Proses menemukan, menganalisis, dan memprioritaskan portofolio aplikasi untuk merencanakan migrasi. Untuk informasi selengkapnya, lihat Mengevaluasi kesiapan migrasi.

predikat

Kondisi kueri yang mengembalikan `true` atau`false`, biasanya terletak di `WHERE` klausa.

predikat pushdown

Teknik optimasi kueri database yang menyaring data dalam kueri sebelum transfer. Ini mengurangi jumlah data yang harus diambil dan diproses dari database relasional, dan meningkatkan kinerja kueri.

kontrol preventif

Kontrol keamanan yang dirancang untuk mencegah suatu peristiwa terjadi. Kontrol ini adalah garis pertahanan pertama untuk membantu mencegah akses tidak sah atau perubahan yang tidak diinginkan ke jaringan Anda. Untuk informasi selengkapnya, lihat Kontrol pencegahan dalam Menerapkan kontrol keamanan pada. AWS

principal

Entitas AWS yang dapat melakukan tindakan dan mengakses sumber daya. Entitas ini biasanya merupakan pengguna root untuk Akun AWS, peran IAM, atau pengguna. Untuk informasi selengkapnya, lihat Prinsip dalam istilah dan konsep Peran dalam dokumentasi IAM.

privasi berdasarkan desain

Pendekatan rekayasa sistem yang memperhitungkan privasi melalui seluruh proses pengembangan.

zona host pribadi

Container yang menyimpan informasi tentang bagaimana Anda ingin Amazon Route 53 merespons kueri DNS untuk domain dan subdomainnya dalam satu atau lebih. VPCs Untuk informasi selengkapnya, lihat Bekerja dengan zona yang dihosting pribadi di dokumentasi Route 53.

kontrol proaktif

Kontrol keamanan yang dirancang untuk mencegah penyebaran sumber daya yang tidak sesuai. Kontrol ini memindai sumber daya sebelum disediakan. Jika sumber daya tidak sesuai dengan kontrol, maka itu tidak disediakan. Untuk informasi selengkapnya, lihat panduan referensi Kontrol dalam AWS Control Tower dokumentasi dan lihat Kontrol proaktif dalam Menerapkan kontrol keamanan pada AWS.

manajemen siklus hidup produk (PLM)

Manajemen data dan proses untuk suatu produk di seluruh siklus hidupnya, mulai dari desain, pengembangan, dan peluncuran, melalui pertumbuhan dan kematangan, hingga penurunan dan penghapusan.

lingkungan produksi

Lihat lingkungan.

pengontrol logika yang dapat diprogram (PLC)

Di bidang manufaktur, komputer yang sangat andal dan mudah beradaptasi yang memantau mesin dan mengotomatiskan proses manufaktur.

rantai cepat

Menggunakan output dari satu prompt LLM sebagai input untuk prompt berikutnya untuk menghasilkan respons yang lebih baik. Teknik ini digunakan untuk memecah tugas yang kompleks menjadi subtugas, atau untuk secara iteratif memperbaiki atau memperluas respons awal. Ini membantu meningkatkan akurasi dan relevansi respons model dan memungkinkan hasil yang lebih terperinci dan dipersonalisasi.

pseudonimisasi

Proses penggantian pengidentifikasi pribadi dalam kumpulan data dengan nilai placeholder. Pseudonimisasi dapat membantu melindungi privasi pribadi. Data pseudonim masih dianggap sebagai data pribadi.

publish/subscribe (pub/sub)

Pola yang memungkinkan komunikasi asinkron antara layanan mikro untuk meningkatkan skalabilitas dan daya tanggap. Misalnya, dalam MES berbasis layanan mikro, layanan mikro dapat mempublikasikan pesan peristiwa ke saluran yang dapat berlangganan layanan mikro lainnya. Sistem dapat menambahkan layanan mikro baru tanpa mengubah layanan penerbitan.

Q

rencana kueri

Serangkaian langkah, seperti instruksi, yang digunakan untuk mengakses data dalam sistem database relasional SQL.

regresi rencana kueri

Ketika pengoptimal layanan database memilih rencana yang kurang optimal daripada sebelum perubahan yang diberikan ke lingkungan database. Hal ini dapat disebabkan oleh perubahan statistik, kendala, pengaturan lingkungan, pengikatan parameter kueri, dan pembaruan ke mesin database.

R

Matriks RACI

Lihat bertanggung jawab, akuntabel, dikonsultasikan, diinformasikan (RACI).

LAP

Lihat Retrieval Augmented Generation.

ransomware

Perangkat lunak berbahaya yang dirancang untuk memblokir akses ke sistem komputer atau data sampai pembayaran dilakukan.

Matriks RASCI

Lihat bertanggung jawab, akuntabel, dikonsultasikan, diinformasikan (RACI).

RCAC

Lihat kontrol akses baris dan kolom.

replika baca

Salinan database yang digunakan untuk tujuan read-only. Anda dapat merutekan kueri ke replika baca untuk mengurangi beban pada database utama Anda.

arsitek ulang

Lihat 7 Rs.

tujuan titik pemulihan (RPO)

Jumlah waktu maksimum yang dapat diterima sejak titik pemulihan data terakhir. Ini menentukan apa yang dianggap sebagai kehilangan data yang dapat diterima antara titik pemulihan terakhir dan gangguan layanan.

tujuan waktu pemulihan (RTO)

Penundaan maksimum yang dapat diterima antara gangguan layanan dan pemulihan layanan.

refactor

Lihat 7 Rs.

Region

Kumpulan AWS sumber daya di wilayah geografis. Masing-masing Wilayah AWS terisolasi dan independen dari yang lain untuk memberikan toleransi kesalahan, stabilitas, dan ketahanan. Untuk informasi selengkapnya, lihat Menentukan Wilayah AWS akun yang dapat digunakan.

regresi

Teknik ML yang memprediksi nilai numerik. Misalnya, untuk memecahkan masalah "Berapa harga rumah ini akan dijual?" Model ML dapat menggunakan model regresi linier untuk memprediksi harga jual rumah berdasarkan fakta yang diketahui tentang rumah (misalnya, luas persegi).

rehost

Lihat 7 Rs.

melepaskan

Dalam proses penyebaran, tindakan mempromosikan perubahan pada lingkungan produksi.

memindahkan

Lihat 7 Rs.

memplatform ulang

Lihat 7 Rs.

pembelian kembali

Lihat 7 Rs.

ketahanan

Kemampuan aplikasi untuk melawan atau pulih dari gangguan. Ketersediaan tinggi dan pemulihan bencana adalah pertimbangan umum ketika merencanakan ketahanan di. AWS Cloud Untuk informasi lebih lanjut, lihat AWS Cloud Ketahanan.

kebijakan berbasis sumber daya

Kebijakan yang dilampirkan ke sumber daya, seperti bucket Amazon S3, titik akhir, atau kunci enkripsi. Jenis kebijakan ini menentukan prinsipal mana yang diizinkan mengakses, tindakan yang didukung, dan kondisi lain yang harus dipenuhi.

matriks yang bertanggung jawab, akuntabel, dikonsultasikan, diinformasikan (RACI)

Matriks yang mendefinisikan peran dan tanggung jawab untuk semua pihak yang terlibat dalam kegiatan migrasi dan operasi cloud. Nama matriks berasal dari jenis tanggung jawab yang

didefinisikan dalam matriks: bertanggung jawab (R), akuntabel (A), dikonsultasikan (C), dan diinformasikan (I). Jenis dukungan (S) adalah opsional. Jika Anda menyertakan dukungan, matriks disebut matriks RASCI, dan jika Anda mengecualikannya, itu disebut matriks RACI.

kontrol responsif

Kontrol keamanan yang dirancang untuk mendorong remediasi efek samping atau penyimpangan dari garis dasar keamanan Anda. Untuk informasi selengkapnya, lihat Kontrol responsif dalam Menerapkan kontrol keamanan pada AWS.

melestarikan

Lihat 7 Rs.

pensiun

Lihat 7 Rs.

Retrieval Augmented Generation (RAG)

Teknologi AI generatif di mana LLM mereferensikan sumber data otoritatif yang berada di luar sumber data pelatihannya sebelum menghasilkan respons. Misalnya, model RAG mungkin melakukan pencarian semantik dari basis pengetahuan organisasi atau data kustom. Untuk informasi lebih lanjut, lihat Apa itu RAG.

rotasi

Proses memperbarui rahasia secara berkala untuk membuatnya lebih sulit bagi penyerang untuk mengakses kredensil.

kontrol akses baris dan kolom (RCAC)

Penggunaan ekspresi SQL dasar dan fleksibel yang telah menetapkan aturan akses. RCAC terdiri dari izin baris dan topeng kolom.

RPO

Lihat tujuan titik pemulihan.

RTO

Lihat tujuan waktu pemulihan.

buku runbook

Satu set prosedur manual atau otomatis yang diperlukan untuk melakukan tugas tertentu. Ini biasanya dibangun untuk merampingkan operasi berulang atau prosedur dengan tingkat kesalahan yang tinggi.

D

SAML 2.0

Standar terbuka yang digunakan oleh banyak penyedia identitas (IdPs). Fitur ini memungkinkan sistem masuk tunggal gabungan (SSO), sehingga pengguna dapat masuk ke Konsol Manajemen AWS atau memanggil operasi AWS API tanpa Anda harus membuat pengguna di IAM untuk semua orang di organisasi Anda. Untuk informasi lebih lanjut tentang federasi berbasis SAMP 2.0, lihat Tentang federasi berbasis SAMP 2.0 dalam dokumentasi IAM.

PENIPUAN

Lihat kontrol pengawasan dan akuisisi data.

SCP

Lihat kebijakan kontrol layanan.

Rahasia

Dalam AWS Secrets Manager, informasi rahasia atau terbatas, seperti kata sandi atau kredensi pengguna, yang Anda simpan dalam bentuk terenkripsi. Ini terdiri dari nilai rahasia dan metadata-nya. Nilai rahasia dapat berupa biner, string tunggal, atau beberapa string. Untuk informasi selengkapnya, lihat Apa yang ada di rahasia Secrets Manager? dalam dokumentasi Secrets Manager.

keamanan dengan desain

Pendekatan rekayasa sistem yang memperhitungkan keamanan melalui seluruh proses pengembangan.

kontrol keamanan

Pagar pembatas teknis atau administratif yang mencegah, mendeteksi, atau mengurangi kemampuan pelaku ancaman untuk mengeksploitasi kerentanan keamanan. Ada empat jenis kontrol keamanan utama: preventif, detektif, responsif, dan proaktif.

pengerasan keamanan

Proses mengurangi permukaan serangan untuk membuatnya lebih tahan terhadap serangan. Ini dapat mencakup tindakan seperti menghapus sumber daya yang tidak lagi diperlukan, menerapkan praktik keamanan terbaik untuk memberikan hak istimewa paling sedikit, atau menonaktifkan fitur yang tidak perlu dalam file konfigurasi.

sistem informasi keamanan dan manajemen acara (SIEM)

Alat dan layanan yang menggabungkan sistem manajemen informasi keamanan (SIM) dan manajemen acara keamanan (SEM). Sistem SIEM mengumpulkan, memantau, dan menganalisis data dari server, jaringan, perangkat, dan sumber lain untuk mendeteksi ancaman dan pelanggaran keamanan, dan untuk menghasilkan peringatan.

otomatisasi respons keamanan

Tindakan yang telah ditentukan dan diprogram yang dirancang untuk secara otomatis merespons atau memulihkan peristiwa keamanan. Otomatisasi ini berfungsi sebagai kontrol keamanan detektif atau responsif yang membantu Anda menerapkan praktik terbaik AWS keamanan. Contoh tindakan respons otomatis termasuk memodifikasi grup keamanan VPC, menambal instans Amazon EC2, atau memutar kredensil.

enkripsi sisi server

Enkripsi data di tujuannya, oleh Layanan AWS yang menerimanya.

kebijakan kontrol layanan (SCP)

Kebijakan yang menyediakan kontrol terpusat atas izin untuk semua akun di organisasi. AWS Organizations SCPs menentukan pagar pembatas atau menetapkan batasan pada tindakan yang dapat didelegasikan oleh administrator kepada pengguna atau peran. Anda dapat menggunakan SCPs daftar izin atau daftar penolakan, untuk menentukan layanan atau tindakan mana yang diizinkan atau dilarang. Untuk informasi selengkapnya, lihat Kebijakan kontrol layanan dalam AWS Organizations dokumentasi.

titik akhir layanan

URL titik masuk untuk file Layanan AWS. Anda dapat menggunakan endpoint untuk terhubung secara terprogram ke layanan target. Untuk informasi selengkapnya, lihat Layanan AWS titik akhir di Referensi Umum AWS.

perjanjian tingkat layanan (SLA)

Perjanjian yang menjelaskan apa yang dijanjikan tim TI untuk diberikan kepada pelanggan mereka, seperti waktu kerja dan kinerja layanan.

indikator tingkat layanan (SLI)

Pengukuran aspek kinerja layanan, seperti tingkat kesalahan, ketersediaan, atau throughputnya.

tujuan tingkat layanan (SLO)

Metrik target yang mewakili kesehatan layanan, yang diukur dengan indikator tingkat layanan.

model tanggung jawab bersama

Model yang menjelaskan tanggung jawab yang Anda bagikan AWS untuk keamanan dan kepatuhan cloud. AWS bertanggung jawab atas keamanan cloud, sedangkan Anda bertanggung jawab atas keamanan di cloud. Untuk informasi selengkapnya, lihat Model tanggung jawab bersama.

SIEM

Lihat informasi keamanan dan sistem manajemen acara.

titik kegagalan tunggal (SPOF)

Kegagalan dalam satu komponen penting dari aplikasi yang dapat mengganggu sistem.

SLA

Lihat perjanjian tingkat layanan.

SLI

Lihat indikator tingkat layanan.

SLO

Lihat tujuan tingkat layanan.

split-and-seed model

Pola untuk menskalakan dan mempercepat proyek modernisasi. Ketika fitur baru dan rilis produk didefinisikan, tim inti berpisah untuk membuat tim produk baru. Ini membantu meningkatkan kemampuan dan layanan organisasi Anda, meningkatkan produktivitas pengembang, dan

mendukung inovasi yang cepat. Untuk informasi lebih lanjut, lihat Pendekatan bertahap untuk memodernisasi aplikasi di. AWS Cloud

SPOF

Lihat satu titik kegagalan.

skema bintang

Struktur organisasi database yang menggunakan satu tabel fakta besar untuk menyimpan data transaksional atau terukur dan menggunakan satu atau lebih tabel dimensi yang lebih kecil untuk menyimpan atribut data. Struktur ini dirancang untuk digunakan dalam gudang data atau untuk tujuan intelijen bisnis.

pola ara pencekik

Pendekatan untuk memodernisasi sistem monolitik dengan menulis ulang secara bertahap dan mengganti fungsionalitas sistem sampai sistem warisan dapat dinonaktifkan. Pola ini menggunakan analogi pohon ara yang tumbuh menjadi pohon yang sudah mapan dan akhirnya mengatasi dan menggantikan inangnya. Pola ini diperkenalkan oleh Martin Fowler sebagai cara untuk mengelola risiko saat menulis ulang sistem monolitik. Untuk contoh cara menerapkan pola ini, lihat Memodernisasi layanan web Microsoft ASP.NET (ASMX) lama secara bertahap menggunakan container dan Amazon API Gateway.

subnet

Rentang alamat IP dalam VPC Anda. Subnet harus berada di Availability Zone tunggal.

kontrol pengawasan dan akuisisi data (SCADA)

Di bidang manufaktur, sistem yang menggunakan perangkat keras dan perangkat lunak untuk memantau aset fisik dan operasi produksi.

enkripsi simetris

Algoritma enkripsi yang menggunakan kunci yang sama untuk mengenkripsi dan mendekripsi data.

pengujian sintetis

Menguji sistem dengan cara yang mensimulasikan interaksi pengguna untuk mendeteksi potensi masalah atau untuk memantau kinerja. Anda dapat menggunakan Amazon CloudWatch Synthetics untuk membuat tes ini.

sistem prompt

Teknik untuk memberikan konteks, instruksi, atau pedoman ke LLM untuk mengarahkan perilakunya. Permintaan sistem membantu mengatur konteks dan menetapkan aturan untuk interaksi dengan pengguna.

T

tag

Pasangan nilai kunci yang bertindak sebagai metadata untuk mengatur sumber daya Anda. AWS Tanda membantu Anda mengelola, mengidentifikasi, mengatur, dan memfilter sumber daya. Untuk informasi selengkapnya, lihat Menandai sumber daya AWS.

variabel target

Nilai yang Anda coba prediksi dalam ML yang diawasi. Ini juga disebut sebagai variabel hasil. Misalnya, dalam pengaturan manufaktur, variabel target bisa menjadi cacat produk.

daftar tugas

Alat yang digunakan untuk melacak kemajuan melalui runbook. Daftar tugas berisi ikhtisar runbook dan daftar tugas umum yang harus diselesaikan. Untuk setiap tugas umum, itu termasuk perkiraan jumlah waktu yang dibutuhkan, pemilik, dan kemajuan.

lingkungan uji

Lihat lingkungan.

pelatihan

Untuk menyediakan data bagi model ML Anda untuk dipelajari. Data pelatihan harus berisi jawaban yang benar. Algoritma pembelajaran menemukan pola dalam data pelatihan yang memetakan atribut data input ke target (jawaban yang ingin Anda prediksi). Ini menghasilkan model ML yang menangkap pola-pola ini. Anda kemudian dapat menggunakan model ML untuk membuat prediksi pada data baru yang Anda tidak tahu targetnya.

gerbang transit

Hub transit jaringan yang dapat Anda gunakan untuk menghubungkan jaringan Anda VPCs dan lokal. Untuk informasi selengkapnya, lihat Apa itu gateway transit dalam AWS Transit Gateway dokumentasi.

alur kerja berbasis batang

Pendekatan di mana pengembang membangun dan menguji fitur secara lokal di cabang fitur dan kemudian menggabungkan perubahan tersebut ke cabang utama. Cabang utama kemudian dibangun untuk pengembangan, praproduksi, dan lingkungan produksi, secara berurutan.

akses tepercaya

Memberikan izin ke layanan yang Anda tentukan untuk melakukan tugas di organisasi Anda di dalam AWS Organizations dan di akunnya atas nama Anda. Layanan tepercaya menciptakan peran terkait layanan di setiap akun, ketika peran itu diperlukan, untuk melakukan tugas manajemen untuk Anda. Untuk informasi selengkapnya, lihat Menggunakan AWS Organizations dengan AWS layanan lain dalam AWS Organizations dokumentasi.

penyetelan

Untuk mengubah aspek proses pelatihan Anda untuk meningkatkan akurasi model ML. Misalnya, Anda dapat melatih model ML dengan membuat set pelabelan, menambahkan label, dan kemudian mengulangi langkah-langkah ini beberapa kali di bawah pengaturan yang berbeda untuk mengoptimalkan model.

tim dua pizza

Sebuah DevOps tim kecil yang bisa Anda beri makan dengan dua pizza. Ukuran tim dua pizza memastikan peluang terbaik untuk berkolaborasi dalam pengembangan perangkat lunak.

U

waswas

Sebuah konsep yang mengacu pada informasi yang tidak tepat, tidak lengkap, atau tidak diketahui yang dapat merusak keandalan model ML prediktif. Ada dua jenis ketidakpastian: ketidakpastian epistemik disebabkan oleh data yang terbatas dan tidak lengkap, sedangkan ketidakpastian aleatorik disebabkan oleh kebisingan dan keacakan yang melekat dalam data.

tugas yang tidak terdiferensiasi

Juga dikenal sebagai angkat berat, pekerjaan yang diperlukan untuk membuat dan mengoperasikan aplikasi tetapi itu tidak memberikan nilai langsung kepada pengguna akhir atau memberikan keunggulan kompetitif. Contoh tugas yang tidak terdiferensiasi termasuk pengadaan, pemeliharaan, dan perencanaan kapasitas.

lingkungan atas

Lihat lingkungan.

V

menyedot debu

Operasi pemeliharaan database yang melibatkan pembersihan setelah pembaruan tambahan untuk merebut kembali penyimpanan dan meningkatkan kinerja.

kendali versi

Proses dan alat yang melacak perubahan, seperti perubahan kode sumber dalam repositori.

Peering VPC

Koneksi antara dua VPCs yang memungkinkan Anda untuk merutekan lalu lintas dengan menggunakan alamat IP pribadi. Untuk informasi selengkapnya, lihat Apa itu peering VPC di dokumentasi VPC Amazon.

kerentanan

Kelemahan perangkat lunak atau perangkat keras yang membahayakan keamanan sistem.

W

cache hangat

Cache buffer yang berisi data terkini dan relevan yang sering diakses. Instance database dapat membaca dari cache buffer, yang lebih cepat daripada membaca dari memori utama atau disk.

data hangat

Data yang jarang diakses. Saat menanyakan jenis data ini, kueri yang cukup lambat biasanya dapat diterima.

fungsi jendela

Fungsi SQL yang melakukan perhitungan pada sekelompok baris yang berhubungan dengan catatan saat ini. Fungsi jendela berguna untuk memproses tugas, seperti menghitung rata-rata bergerak atau mengakses nilai baris berdasarkan posisi relatif dari baris saat ini.

beban kerja

Kumpulan sumber daya dan kode yang memberikan nilai bisnis, seperti aplikasi yang dihadapi pelanggan atau proses backend.

aliran kerja

Grup fungsional dalam proyek migrasi yang bertanggung jawab atas serangkaian tugas tertentu. Setiap alur kerja independen tetapi mendukung alur kerja lain dalam proyek. Misalnya, alur kerja portofolio bertanggung jawab untuk memprioritaskan aplikasi, perencanaan gelombang, dan mengumpulkan metadata migrasi. Alur kerja portofolio mengirimkan aset ini ke alur kerja migrasi, yang kemudian memigrasikan server dan aplikasi.

CACING

Lihat menulis sekali, baca banyak.

WQF

Lihat AWS Kerangka Kualifikasi Beban Kerja.

tulis sekali, baca banyak (WORM)

Model penyimpanan yang menulis data satu kali dan mencegah data dihapus atau dimodifikasi. Pengguna yang berwenang dapat membaca data sebanyak yang diperlukan, tetapi mereka tidak dapat mengubahnya. Infrastruktur penyimpanan data ini dianggap tidak dapat diubah.

Z

eksploitasi zero-day

Serangan, biasanya malware, yang memanfaatkan kerentanan zero-day.

kerentanan zero-day

Cacat atau kerentanan yang tak tanggung-tanggung dalam sistem produksi. Aktor ancaman dapat menggunakan jenis kerentanan ini untuk menyerang sistem. Pengembang sering menyadari kerentanan sebagai akibat dari serangan tersebut.

bisikan zero-shot

Memberikan LLM dengan instruksi untuk melakukan tugas tetapi tidak ada contoh (tembakan) yang dapat membantu membimbingnya. LLM harus menggunakan pengetahuan pra-terlatih untuk

menangani tugas. Efektivitas bidikan nol tergantung pada kompleksitas tugas dan kualitas prompt. Lihat juga beberapa bidikan yang diminta.

aplikasi zombie

Aplikasi yang memiliki CPU rata-rata dan penggunaan memori di bawah 5 persen. Dalam proyek migrasi, adalah umum untuk menghentikan aplikasi ini.

Terjemahan disediakan oleh mesin penerjemah. Jika konten terjemahan yang diberikan bertentangan dengan versi bahasa Inggris aslinya, utamakan versi bahasa Inggris.