AWS Dasar-dasar multi-wilayah

AWS Bimbingan Preskriptif

AWS Bimbingan Preskriptif: AWS Dasar-dasar multi-wilayah

Table of Contents

AWS Dasar-dasar multi-wilayah

John Formento, Amazon Web Services ()AWS

September 2025 (riwayat dokumen)

Panduan tingkat 300 tingkat lanjut ini ditujukan untuk arsitek cloud dan pemimpin senior yang membangun beban kerja AWS dan tertarik menggunakan arsitektur Multi-wilayah untuk meningkatkan ketahanan beban kerja mereka. Panduan ini mengasumsikan pengetahuan dasar tentang AWS infrastruktur dan layanan. Ini menguraikan kasus penggunaan Multi-wilayah yang umum, berbagi konsep dan implikasi Multi-wilayah mendasar seputar desain, pengembangan, dan penerapan, dan memberikan panduan preskriptif untuk membantu Anda menentukan dengan lebih baik apakah arsitektur Multi-wilayah tepat untuk beban kerja Anda.

Dalam panduan ini:

Apakah Anda sudah Well-Architected?

AWS Well-Architected Framework membantu Anda memahami pro dan kontra dari keputusan yang Anda buat saat membangun sistem di cloud. Enam pilar Kerangka memberikan praktik terbaik arsitektur untuk merancang dan mengoperasikan sistem yang andal, aman, efisien, hemat biaya, dan berkelanjutan. Anda dapat menggunakan AWS Well-Architected Tool, yang tersedia tanpa biaya pada Konsol Manajemen AWS, untuk meninjau beban kerja Anda terhadap praktik terbaik ini dengan menjawab serangkaian pertanyaan untuk setiap pilar.

Untuk panduan ahli tambahan dan praktik terbaik untuk arsitektur cloud Anda, termasuk penerapan arsitektur referensi, diagram, dan panduan teknis, lihat Pusat Arsitektur.AWS

Pengantar

Masing-masing AWS Regionterdiri dari beberapa Availability Zone yang independen dan terpisah secara fisik dalam suatu wilayah geografis. Pemisahan logis yang ketat antara layanan perangkat lunak di setiap Wilayah dipertahankan. Desain yang bertujuan ini memastikan bahwa kegagalan infrastruktur atau layanan di satu Wilayah tidak mengakibatkan kegagalan yang berkorelasi di Wilayah lain.

Sebagian besar AWS pengguna dapat mencapai tujuan ketahanan mereka untuk beban kerja di satu Wilayah dengan menggunakan beberapa Availability Zone atau Regional. Layanan AWS Namun, sebagian pengguna mengejar arsitektur Multi-wilayah karena tiga alasan:

- Mereka memiliki ketersediaan tinggi dan kontinuitas persyaratan operasi untuk beban kerja tingkat tertinggi mereka dan ingin menetapkan waktu pemulihan terbatas dari gangguan yang berdampak pada sumber daya di satu Wilayah.
- Mereka perlu memenuhi persyaratan kedaulatan data (seperti kepatuhan terhadap hukum, peraturan, dan kepatuhan setempat) yang memerlukan beban kerja untuk beroperasi dalam yurisdiksi tertentu.
- Mereka perlu meningkatkan kinerja dan pengalaman pelanggan untuk beban kerja dengan menjalankan beban kerja di lokasi yang paling dekat dengan pengguna akhir mereka.

Panduan ini berfokus pada ketersediaan tinggi dan kontinuitas persyaratan operasi, dan membantu Anda menavigasi pertimbangan untuk mengadopsi arsitektur Multi-wilayah untuk beban kerja. Ini menjelaskan konsep dasar yang berlaku untuk desain, pengembangan, dan penyebaran beban kerja Multi-wilayah, dan menyediakan kerangka kerja preskriptif untuk membantu Anda menentukan apakah arsitektur Multi-wilayah adalah pilihan yang tepat untuk beban kerja tertentu. Anda perlu memastikan bahwa arsitektur Multi-region adalah pilihan yang tepat untuk beban kerja Anda karena arsitektur ini menantang, dan jika arsitektur Multi-region tidak dibangun dengan benar, ketersediaan beban kerja secara keseluruhan mungkin berkurang.

Rekayasa dan operasi untuk ketahanan di satu Wilayah

Sebelum Anda menyelami konsep Multi-wilayah, mulailah dengan mengonfirmasi bahwa beban kerja Anda sudah sekuat mungkin di satu Wilayah. Untuk mencapai hal ini, evaluasi beban kerja Anda terhadap pilar keandalan dan pilar keunggulan operasional dari AWS Well-Architected Framework, dan buat perubahan yang diperlukan berdasarkan trade-off dan penilaian risiko. Konsep-konsep berikut tercakup dalam AWS Well-Architected Framework:

- Segmentasi beban kerja berdasarkan batas domain
- Kontrak layanan yang terdefinisi dengan baik
- Manajemen ketergantungan dan kopling
- Menangani kegagalan, percobaan ulang, dan strategi mundur
- Operasi idempoten dan transaksi stateful versus stateless
- Kesiapan operasional dan manajemen perubahan
- Memahami kesehatan beban kerja
- Menanggapi peristiwa

Untuk meningkatkan ketahanan wilayah Tunggal, tinjau dan terapkan konsep yang dibahas dalam paper Advanced Multi-AZ Resilience Patterns: Detecting and Mitigating Grey Failures. Paper ini memberikan praktik terbaik untuk menggunakan replika di setiap Availability Zone untuk menampung kegagalan dan memperluas konsep Multi-AZ yang diperkenalkan dalam AWS Well Architected Framework. Meskipun arsitektur Multi-wilayah dapat mengurangi mode kegagalan yang terikat pada Availability Zone, ada trade-off yang datang dengan pendekatan Multi-region yang harus Anda pertimbangkan. Itulah sebabnya kami menyarankan Anda memulai dengan pendekatan Multi-AZ, dan kemudian mengevaluasi beban kerja tertentu terhadap dasar-dasar arsitektur Multi-wilayah untuk menentukan apakah pendekatan Multi-wilayah dapat meningkatkan ketahanan beban kerja.

Multi-Region fundamental 1: Memahami persyaratan

Seperti disebutkan sebelumnya, ketersediaan dan kontinuitas operasi yang tinggi adalah alasan umum untuk mengejar arsitektur Multi-wilayah. Metrik ketersediaan mengukur persentase waktu beban kerja tersedia untuk digunakan selama periode tertentu, sedangkan metrik kontinuitas operasi mengukur waktu pemulihan untuk peristiwa durasi skala besar, dan biasanya lebih lama.

Mengukur ketersediaan adalah proses yang hampir berkelanjutan. Pengukuran spesifik dapat bervariasi tetapi biasanya menyatu di sekitar metrik ketersediaan target, paling sering disebut sebagai sembilan (seperti ketersediaan 99,99 persen). Dengan tujuan ketersediaan, satu ukuran tidak cocok untuk semua. Anda harus menetapkan sasaran ketersediaan pada tingkat beban kerja dan memisahkan komponen non-kritis dari komponen penting, alih-alih menerapkan satu tujuan di semua beban kerja.

Untuk kelangsungan operasi, point-in-time pengukuran berikut biasanya digunakan:

- Tujuan waktu pemulihan (RTO) - RTO adalah penundaan maksimum yang dapat diterima antara gangguan layanan dan pemulihan layanan. Nilai ini menentukan durasi yang dapat diterima di mana layanan terganggu.
- Tujuan titik pemulihan (RPO) — RPO adalah 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 terbaru dan gangguan layanan.

Mirip dengan menetapkan tujuan ketersediaan, RTO dan RPO juga harus didefinisikan pada tingkat beban kerja. Kontinuitas operasi yang lebih agresif atau ketersediaan yang tinggi membutuhkan peningkatan investasi. Meskipun demikian, tidak setiap aplikasi dapat menuntut atau membutuhkan tingkat ketahanan yang sama. Menyelaraskan pemilik bisnis dan TI untuk menilai kekritisan aplikasi berdasarkan dampak bisnis dan kemudian meningkatkannya dapat membantu memberikan titik awal. Tabel berikut memberikan contoh tiering.

Tabel ini menunjukkan contoh tiering ketahanan untuk perjanjian tingkat layanan (). SLAs

Tingkat ketahanan	Ketersediaan SLA	Waktu henti/tahun yang dapat diterima
Platinum	99,99%	52.60 menit

Tingkat ketahanan	Ketersediaan SLA	Waktu henti/tahun yang dapat diterima
Emas	99,90%	8.77 jam
Perak	99,5%	1.83 hari

Tabel berikut menunjukkan contoh tiering ketahanan untuk RTO dan RPO.

Tingkat ketahanan	RTO Maksimum	RPO Maksimum	Kriteria	Biaya
Platinum	15 menit	5 menit	Beban kerja kritis misi	$$$
Emas	15 menit — 6 jam	2 jam	Beban kerja penting tetapi tidak penting untuk misi	$$
Perak	6 jam — beberapa hari	24 jam	Beban kerja yang tidak kritis	$

Saat Anda merancang beban kerja untuk ketahanan, pertimbangkan hubungan antara ketersediaan tinggi dan kontinuitas operasi. Misalnya, jika beban kerja membutuhkan ketersediaan 99,99 persen, tidak lebih dari 53 menit downtime per tahun dapat ditoleransi. Diperlukan setidaknya 5 menit untuk mendeteksi kegagalan dan 10 menit lagi bagi operator untuk terlibat, membuat keputusan tentang langkah-langkah pemulihan, dan melakukan langkah-langkah ini. Bukan hal yang aneh untuk mengambil 30 hingga 45 menit untuk pulih dari satu masalah. Dalam hal ini, akan bermanfaat untuk memiliki strategi Multi-wilayah untuk memberikan contoh terisolasi yang menghilangkan dampak yang berkorelasi. Hal ini memungkinkan operasi lanjutan dengan gagal dalam waktu terbatas saat Anda melakukan triase penurunan nilai awal secara independen. Di sinilah menentukan waktu pemulihan terbatas yang sesuai dan memastikan ada keselarasan diperlukan.

Pendekatan multi-wilayah mungkin sesuai untuk beban kerja mission-critical yang memiliki kebutuhan ketersediaan ekstrim (misalnya, 99,99 persen atau ketersediaan lebih tinggi) atau kontinuitas ketat persyaratan operasi yang hanya dapat dipenuhi dengan gagal masuk ke Wilayah lain. Namun,

persyaratan ini biasanya hanya berlaku untuk sebagian kecil dari portofolio beban kerja perusahaan yang memiliki waktu pemulihan terbatas yang diukur dalam menit atau jam. Kecuali aplikasi memerlukan waktu pemulihan beberapa menit atau beberapa jam, mungkin pendekatan yang lebih baik untuk menunggu gangguan Regional terhadap aplikasi untuk diperbaiki di Wilayah yang terkena dampak. Pendekatan ini biasanya selaras dengan beban kerja tingkat yang lebih rendah.

Sebelum menerapkan arsitektur Multi-region, pengambil keputusan bisnis dan tim teknis harus selaras dengan implikasi biaya, termasuk penggerak biaya operasional dan infrastruktur. Arsitektur Multi-wilayah yang khas dapat dikenakan biaya yang dua kali lebih besar dari pendekatan Single-region. Meskipun ada beberapa pola Multi-wilayah untuk kelangsungan bisnis, seperti berjalan dengan siaga panas, siaga hangat, atau lampu pilot, pola dengan risiko terendah untuk memenuhi tujuan pemulihan akan melibatkan menjalankan siaga panas, dan akan menggandakan biaya untuk beban kerja Anda.

Panduan utama

- Ketersediaan dan kelangsungan tujuan operasi seperti RTO dan RPO harus ditetapkan per beban kerja dan selaras dengan pemangku kepentingan bisnis dan TI.
- Sebagian besar ketersediaan dan kelangsungan tujuan operasi dapat dipenuhi dalam satu Wilayah. Untuk tujuan yang tidak dapat dipenuhi dalam satu Wilayah, pertimbangkan Multi-wilayah dengan pandangan yang jelas tentang trade-off antara biaya, kompleksitas, dan manfaat.

Multi-Region fundamental 2: Memahami data

Mengelola data adalah masalah yang tidak sepele ketika Anda mengadopsi arsitektur Multi-region. Jarak geografis antar Wilayah memberlakukan latensi yang tidak dapat dihindari yang bermanifestasi sebagai waktu yang diperlukan untuk mereplikasi data di seluruh Wilayah. Pertukaran antara ketersediaan, konsistensi data, dan memperkenalkan latensi yang lebih tinggi ke dalam beban kerja yang menggunakan arsitektur Multi-wilayah akan diperlukan. Apakah Anda menggunakan replikasi asinkron atau sinkron, Anda perlu memodifikasi aplikasi Anda untuk menangani perubahan perilaku yang diterapkan teknologi replikasi. Tantangan seputar konsistensi dan latensi data membuatnya sangat sulit untuk mengambil aplikasi yang sudah ada yang dirancang untuk satu Wilayah dan menjadikannya Multi-wilayah. Memahami persyaratan konsistensi data dan pola akses data untuk beban kerja tertentu sangat penting untuk menimbang trade-off.

2.a: Memahami persyaratan konsistensi data

Teorema CAP memberikan referensi untuk alasan tentang trade-off antara konsistensi data, ketersediaan, dan partisi jaringan. Hanya dua dari persyaratan ini yang dapat dipenuhi pada saat yang sama untuk beban kerja. Menurut definisi, arsitektur Multi-region mencakup partisi jaringan antar Wilayah, jadi Anda harus memilih antara ketersediaan dan konsistensi.

Jika Anda memilih ketersediaan data di seluruh Wilayah, Anda tidak akan mengalami latensi yang signifikan selama operasi penulisan transaksional, karena ketergantungan pada replikasi asinkron dari data komit antar Wilayah menghasilkan pengurangan konsistensi di seluruh Wilayah hingga replikasi selesai. Dengan replikasi asinkron, ketika ada kegagalan di Wilayah primer, ada kemungkinan besar bahwa operasi penulisan akan menunggu replikasi dari Wilayah primer. Hal ini menyebabkan skenario di mana data terbaru tidak tersedia sampai replikasi dilanjutkan, dan proses rekonsiliasi diperlukan untuk menangani transaksi dalam penerbangan yang tidak mereplikasi dari Wilayah yang mengalami gangguan. Skenario ini membutuhkan pemahaman logika bisnis Anda dan menciptakan proses khusus untuk memutar ulang transaksi atau membandingkan penyimpanan data antar Wilayah.

Untuk beban kerja yang disukai replikasi asinkron, Anda dapat menggunakan layanan seperti Amazon Aurora dan Amazon DynamoDB untuk replikasi lintas wilayah asinkron. Baik database global Amazon Aurora dan tabel global Amazon DynamoDB memiliki metrik CloudWatchAmazon default untuk membantu memantau kelambatan replikasi. Database global Aurora terdiri dari satu Wilayah utama tempat data Anda ditulis, dan hingga lima Wilayah sekunder hanya-baca. Tabel global

DynamoDB terdiri dari tabel replika multi-aktif di sejumlah Wilayah tempat data Anda ditulis dan dibaca.

Rekayasa beban kerja untuk memanfaatkan arsitektur berbasis peristiwa merupakan manfaat bagi strategi Multi-wilayah, karena itu berarti bahwa beban kerja dapat merangkul replikasi data asinkron dan memungkinkan rekonstruksi status dengan memutar ulang peristiwa. Karena layanan streaming dan pesan menyangga data payload pesan dalam satu Wilayah, proses failover atau failback Regional harus menyertakan mekanisme untuk mengarahkan aliran data masukan klien. Proses ini juga harus mendamaikan muatan dalam penerbangan atau tidak terkirim yang disimpan di Wilayah yang mengalami gangguan tersebut.

Jika Anda memilih persyaratan konsistensi CAP dan menggunakan database yang direplikasi secara sinkron di seluruh Wilayah untuk mendukung aplikasi Anda yang berjalan secara bersamaan dari beberapa Wilayah, Anda menghapus risiko kehilangan data dan menjaga agar data tetap sinkron antar Wilayah. Namun, ini memperkenalkan karakteristik latensi yang lebih tinggi, karena penulisan perlu berkomitmen untuk lebih dari satu Wilayah, dan Wilayah dapat berjarak ratusan atau ribuan mil dari satu sama lain. Anda perlu memperhitungkan karakteristik latensi ini dalam desain aplikasi Anda. Selain itu, replikasi sinkron dapat memperkenalkan peluang kegagalan yang berkorelasi karena penulisan perlu berkomitmen pada lebih dari satu Wilayah agar berhasil. Jika ada gangguan dalam satu Wilayah, Anda perlu membentuk kuorum agar penulisan berhasil. Ini biasanya melibatkan pengaturan database Anda di tiga Wilayah dan membuat kuorum dua dari tiga Wilayah. Teknologi seperti Paxos dapat membantu mereplikasi dan melakukan data secara serempak tetapi membutuhkan investasi pengembang yang signifikan.

Ketika penulisan melibatkan replikasi sinkron di beberapa Wilayah untuk memenuhi persyaratan konsistensi yang kuat, latensi tulis meningkat dengan urutan besarnya. Latensi tulis yang lebih tinggi bukanlah sesuatu yang biasanya dapat Anda perbaiki ke dalam aplikasi tanpa perubahan signifikan, seperti meninjau kembali strategi timeout dan coba lagi untuk aplikasi Anda. Idealnya, itu harus dipertimbangkan ketika aplikasi pertama kali dirancang. Untuk beban kerja multi-wilayah di mana replikasi sinkron adalah prioritas,AWS Partner solusi dapat membantu.

2.b: Memahami pola akses data

Pola akses data beban kerja bersifat intensif baca atau intensif tulis. Memahami karakteristik ini untuk beban kerja tertentu akan membantu Anda memilih arsitektur Multi-wilayah yang sesuai.

Untuk beban kerja intensif baca seperti konten statis yang sepenuhnya hanya-baca, Anda dapat mencapai arsitektur Multi-wilayah aktif-aktif yang memiliki kompleksitas rekayasa lebih sedikit

jika dibandingkan dengan beban kerja intensif tulis. Menyajikan konten statis di tepi dengan menggunakan jaringan pengiriman konten (CDN) memastikan ketersediaan dengan menyimpan konten yang paling dekat dengan pengguna akhir; menggunakan set fitur seperti failover asal di Amazon CloudFront dapat membantu mencapai hal ini. Pilihan lainnya adalah menerapkan stateless compute di beberapa Wilayah dan menggunakan DNS untuk merutekan pengguna ke Wilayah terdekat untuk membaca konten. Anda dapat menggunakan Amazon Route 53 dengan kebijakan perutean geolokasi untuk mencapai hal ini.

Untuk beban kerja intensif baca yang memiliki persentase lalu lintas baca yang lebih besar daripada lalu lintas tulis, Anda dapat menggunakan strategi baca lokal, tulis global. Ini berarti bahwa semua permintaan tulis masuk ke database di Wilayah tertentu, data direplikasi secara asinkron ke semua Wilayah lain, dan pembacaan dapat dilakukan di Wilayah mana pun. Pendekatan ini membutuhkan beban kerja untuk merangkul konsistensi akhirnya, karena pembacaan lokal mungkin menjadi basi sebagai akibat dari peningkatan latensi untuk replikasi penulisan lintas wilayah.

Database global Aurora dapat membantu penyediaan replika baca di Wilayah siaga yang hanya dapat menangani semua lalu lintas baca secara lokal, dan menyediakan satu penyimpanan data utama di Wilayah tertentu untuk menangani lalu lintas tulis. Data direplikasi secara asinkron dari database utama ke database siaga (baca replika), dan database siaga dapat dipromosikan ke primer jika Anda perlu gagal dalam operasi ke Wilayah siaga. Anda juga dapat menggunakan DynamoDB dalam pendekatan ini. Tabel global DynamoDB dapat menyediakan tabel replika di seluruh Wilayah yang dapat setiap skala untuk mendukung volume lalu lintas baca atau tulis lokal. Ketika aplikasi menulis data ke tabel replika di satu wilayah, DynamoDB otomatis menyebarkan aktivitas tulis tersebut ke tabel replika lain di Wilayah lain. Dengan konfigurasi ini, data direplikasi secara asinkron dari Wilayah primer yang ditentukan ke tabel replika di Wilayah siaga. Tabel replika di Wilayah mana pun selalu dapat menerima penulisan, jadi mempromosikan Region siaga ke primer dikelola di tingkat aplikasi. Sekali lagi, beban kerja harus mencakup konsistensi akhirnya, yang mungkin mengharuskannya ditulis ulang jika tidak dirancang untuk ini sejak awal.

Untuk beban kerja intensif tulis, Wilayah utama harus dipilih dan kemampuan untuk gagal ke Wilayah siaga harus direkayasa ke dalam beban kerja. Dibandingkan dengan pendekatan aktif-aktif, pendekatan siaga primer memiliki trade-off tambahan. Ini karena untuk arsitektur aktif-aktif, beban kerja harus ditulis ulang untuk menangani perutean cerdas ke Wilayah, membangun afinitas sesi, memastikan transaksi idempoten, dan menangani potensi konflik.

Sebagian besar beban kerja yang menggunakan pendekatan Multi-wilayah untuk ketahanan tidak memerlukan pendekatan aktif-aktif. Anda dapat menggunakan strategi sharding untuk memberikan peningkatan ketahanan dengan membatasi ruang lingkup dampak penurunan nilai di seluruh basis

klien. Jika Anda dapat secara efektif menghancurkan basis klien, Anda dapat memilih Wilayah primer yang berbeda untuk setiap pecahan. Misalnya, Anda dapat membelah klien sehingga setengah dari klien selaras dengan Wilayah satu dan setengahnya selaras dengan Wilayah dua. Dengan memperlakukan Regions sebagai sel, Anda dapat membuat pendekatan sel Multi-region, yang menghasilkan pengurangan cakupan dampak untuk beban kerja Anda. Untuk informasi selengkapnya, lihat presentasi AWS re:Invent tentang pendekatan ini.

Anda dapat menggabungkan pendekatan sharding dengan pendekatan siaga utama untuk memberikan kemampuan failover untuk pecahan. Anda perlu merekayasa proses failover yang diuji ke dalam beban kerja dan proses untuk rekonsiliasi data juga, untuk memastikan konsistensi transaksional dari penyimpanan data setelah failover. Ini dibahas secara lebih rinci nanti dalam panduan ini.

Panduan utama

- Ada kemungkinan besar bahwa penulisan yang tertunda untuk replikasi tidak akan dilakukan ke Wilayah siaga ketika ada kegagalan. Data tidak akan tersedia sampai replikasi dilanjutkan (dengan asumsi replikasi asinkron).
- Sebagai bagian dari failover, proses rekonsiliasi data akan diperlukan untuk memastikan bahwa status yang konsisten secara transaksional dipertahankan untuk penyimpanan data yang menggunakan replikasi asinkron. Ini membutuhkan logika bisnis tertentu dan bukan sesuatu yang ditangani oleh penyimpanan data itu sendiri.
- Ketika konsistensi yang kuat diperlukan, beban kerja perlu dimodifikasi untuk mentolerir latensi yang diperlukan dari penyimpanan data yang mereplikasi secara sinkron.

Multi-Region fundamental 3: Memahami dependensi beban kerja Anda

Beban kerja tertentu mungkin memiliki beberapa dependensi di Wilayah, seperti Layanan AWS digunakan, dependensi internal, dependensi pihak ketiga, dependensi jaringan, sertifikat, kunci, rahasia, dan parameter. Untuk memastikan pengoperasian beban kerja selama skenario kegagalan, seharusnya tidak ada ketergantungan antara Wilayah utama dan Wilayah siaga; masing-masing harus dapat beroperasi secara independen dari yang lain. Untuk mencapai hal ini, teliti semua dependensi dalam beban kerja untuk memastikan bahwa mereka tersedia di setiap Wilayah. Hal ini diperlukan karena kegagalan di Wilayah primer seharusnya tidak mempengaruhi Wilayah siaga. Selain itu, Anda harus memahami bagaimana beban kerja beroperasi ketika ketergantungan berada dalam keadaan terdegradasi atau sama sekali tidak tersedia, sehingga Anda dapat merekayasa solusi untuk menangani ini dengan tepat.

3.a: Layanan AWS

Saat Anda merancang arsitektur Multi-wilayah, penting untuk memahami apa Layanan AWS yang akan digunakan, fitur Multi-wilayah dari layanan tersebut, dan solusi apa yang Anda perlukan untuk merekayasa untuk mencapai tujuan Multi-wilayah. Misalnya, Amazon Aurora dan Amazon DynamoDB dapat mereplikasi data secara asinkron ke Wilayah siaga. Semua Layanan AWS dependensi harus tersedia di semua Wilayah tempat beban kerja akan dijalankan. Untuk mengonfirmasi bahwa layanan yang Anda gunakan tersedia di Wilayah yang diinginkan, tinjau daftar Layanan AWS berdasarkan Wilayah.

3.b: Ketergantungan internal dan pihak ketiga

Pastikan bahwa setiap dependensi internal beban kerja tersedia di Wilayah tempat mereka beroperasi. Misalnya, jika beban kerja terdiri dari banyak layanan mikro, identifikasi semua layanan mikro yang terdiri dari kemampuan bisnis dan verifikasi bahwa semua layanan mikro tersebut digunakan di setiap Wilayah tempat beban kerja beroperasi. Atau, tentukan strategi untuk menangani layanan mikro dengan anggun yang menjadi tidak tersedia.

Panggilan Lintas Wilayah antara layanan mikro dalam beban kerja tidak disarankan, dan isolasi Regional harus dipertahankan. Ini karena membuat dependensi lintas wilayah menambah risiko kegagalan yang berkorelasi, yang mengimbangi manfaat implementasi Regional yang terisolasi dari beban kerja. Dependensi lokal mungkin menjadi bagian dari beban kerja juga, jadi penting

untuk memahami bagaimana karakteristik integrasi ini dapat berubah jika Wilayah utama berubah. Misalnya, jika Wilayah siaga terletak lebih jauh dari lingkungan lokal, peningkatan latensi mungkin berdampak negatif.

Memahami solusi perangkat lunak sebagai layanan (SaaS), kit pengembangan perangkat lunak (SDKs), dan dependensi produk pihak ketiga lainnya, dan mampu menjalankan skenario di mana dependensi ini terdegradasi atau tidak tersedia akan memberikan lebih banyak wawasan tentang bagaimana rantai sistem beroperasi dan berperilaku di bawah mode kegagalan yang berbeda. Dependensi ini dapat berada dalam kode aplikasi Anda, seperti mengelola rahasia secara eksternal dengan menggunakan AWS Secrets Manager, atau mereka dapat melibatkan solusi vault pihak ketiga (seperti HashiCorp), atau sistem otentikasi yang memiliki ketergantungan untuk login federasi. AWS IAM Identity Center

Memiliki redundansi dalam hal dependensi dapat meningkatkan ketahanan. Jika solusi SaaS atau ketergantungan pihak ketiga menggunakan primer AWS Region yang sama dengan beban kerja, bekerjalah dengan vendor untuk menentukan apakah postur ketahanan mereka sesuai dengan kebutuhan Anda untuk beban kerja.

Selain itu, waspadai nasib bersama antara beban kerja dan dependensinya, seperti aplikasi pihak ketiga. Jika dependensi tidak tersedia di (atau dari) Wilayah sekunder setelah failover, beban kerja mungkin tidak pulih sepenuhnya.

3.c: Mekanisme failover

DNS biasanya digunakan sebagai mekanisme failover untuk mengalihkan lalu lintas dari Wilayah utama ke Wilayah siaga. Tinjau dan teliti secara kritis semua dependensi yang dibutuhkan mekanisme failover. Misalnya, jika beban kerja Anda menggunakan Amazon Route 53, memahami bahwa bidang kontrol di-host `us-east-1` berarti Anda mengambil ketergantungan pada bidang kontrol di Wilayah tertentu. Ini tidak direkomendasikan sebagai bagian dari mekanisme failover jika Wilayah primer juga `us-east-1` karena menciptakan satu titik kegagalan. Jika Anda menggunakan mekanisme failover lain, Anda harus memiliki pemahaman mendalam tentang skenario di mana failover tidak akan berfungsi seperti yang diharapkan, dan kemudian merencanakan kemungkinan atau mengembangkan mekanisme baru jika diperlukan. Sakelar Wilayah Amazon Application Recovery Controller (ARC) adalah layanan pemulihan Multi-wilayah yang dikelola sepenuhnya yang dapat Anda gunakan sebagai mekanisme failover Anda.

Sebagaimana dibahas di bagian sebelumnya, semua layanan mikro yang merupakan bagian dari kemampuan bisnis harus tersedia di setiap Wilayah di mana beban kerja digunakan. Sebagai

bagian dari strategi failover, semua layanan mikro yang merupakan bagian dari kemampuan bisnis harus gagal bersama-sama untuk menghilangkan kemungkinan panggilan lintas wilayah. Atau, jika layanan mikro gagal secara independen, ada potensi perilaku yang tidak diinginkan seperti layanan mikro yang berpotensi melakukan panggilan lintas wilayah. Ini memperkenalkan latensi dan dapat menyebabkan beban kerja menjadi tidak tersedia selama batas waktu klien.

3.d: Ketergantungan konfigurasi

Sertifikat, kunci, rahasia, Amazon Machine Images (AMIs), gambar kontainer, dan parameter adalah bagian dari analisis dependensi yang diperlukan saat merancang untuk arsitektur Multi-region. Jika memungkinkan, yang terbaik adalah melokalkan komponen-komponen ini di setiap Wilayah sehingga mereka tidak memiliki nasib bersama antar Wilayah untuk dependensi ini. Misalnya, Anda harus memvariasikan tanggal kedaluwarsa sertifikat untuk mencegah skenario di mana sertifikat kedaluwarsa (dengan alarm disetel ke “beri tahu sebelumnya”) berdampak pada beberapa Wilayah.

Kunci enkripsi dan rahasia harus spesifik Wilayah juga. Dengan begitu, jika ada kesalahan dalam rotasi kunci atau rahasia, dampaknya terbatas pada Wilayah tertentu.

Terakhir, parameter beban kerja apa pun harus disimpan secara lokal agar beban kerja dapat diambil di Wilayah tertentu.

Panduan utama

- Arsitektur multi-wilayah mendapat manfaat dari pemisahan fisik dan logis antar Wilayah. Memperkenalkan dependensi lintas wilayah pada lapisan aplikasi merusak manfaat ini. Hindari ketergantungan seperti itu.
- Kontrol failover harus berfungsi tanpa dependensi pada Wilayah utama.
- Failover harus dikoordinasikan di seluruh perjalanan pengguna untuk menghapus kemungkinan peningkatan latensi dan ketergantungan panggilan lintas wilayah.

Multi-Region fundamental 4: Kesiapan operasional

Mengoperasikan beban kerja Multi-wilayah adalah tugas kompleks yang disertai dengan tantangan operasional yang khusus untuk arsitektur Multi-wilayah. Ini termasuk Akun AWS manajemen, proses penerapan retooled, menciptakan strategi observabilitas Multi-wilayah, membuat dan menguji proses pemulihan, dan kemudian mengelola biaya. Tinjauan Kesiapan Operasional (ORR) dapat membantu tim menyiapkan beban kerja untuk produksi, baik itu berjalan di satu Wilayah atau di beberapa Wilayah.

4.a: manajemen Akun AWS

Untuk menerapkan beban kerja di seluruh wilayah Wilayah AWS, pastikan ada paritas di semua Layanan AWS kuota dalam akun di seluruh Wilayah. Pertama, identifikasi semua Layanan AWS yang merupakan bagian dari arsitektur, lihat penggunaan yang direncanakan di Wilayah siaga, dan kemudian bandingkan penggunaan yang direncanakan dengan penggunaan saat ini. Dalam beberapa kasus, jika Wilayah siaga belum pernah digunakan sebelumnya, Anda dapat mereferensikan kuota layanan default untuk memahami titik awal. Kemudian, di seluruh layanan yang akan digunakan, mintalah peningkatan kuota dengan menggunakan konsol Service Quotas (diperlukan login) atau. APIs

Konfigurasikan peran AWS Identity and Access Management (IAM) di setiap Wilayah untuk memberikan operator, perkakas otomatisasi, dan izin Layanan AWS yang sesuai untuk sumber daya dalam Wilayah siaga. Untuk mencapai isolasi Regional untuk arsitektur Multi-region, isolasi peran berdasarkan Wilayah. Pastikan izin sudah ada sebelum ditayangkan dengan Wilayah siaga.

4.b: Praktik penyebaran

Kemampuan Multi-Region dapat mempersulit penerapan beban kerja ke beberapa Wilayah. Anda perlu memastikan bahwa Anda menyebarkan ke satu Wilayah pada satu waktu. Misalnya, jika Anda menggunakan pendekatan aktif-pasif, Anda harus menerapkan ke Wilayah utama terlebih dahulu dan kemudian ke Wilayah siaga. AWS CloudFormationmembantu Anda menyebarkan infrastruktur ke satu atau beberapa Wilayah, dan dapat disesuaikan sesuai dengan kebutuhan Anda. AWS CodePipelinembantu Anda membangun pipeline integration/continuous pengiriman berkelanjutan (CI/CD), yang memiliki tindakan lintas wilayah yang memungkinkan penerapan ke Wilayah yang berbeda dari Wilayah tempat pipa berada. Ini, dikombinasikan dengan strategi penerapan yang kuat seperti biru/hijau, memungkinkan penerapan downtime minimum hingga nol.

Namun, penyebaran kemampuan stateful dapat menjadi lebih kompleks ketika status aplikasi atau data tidak dieksternalisasi ke penyimpanan persisten. Dalam situasi ini, sesuaikan proses penyebaran dengan hati-hati agar sesuai dengan kebutuhan Anda. Rancang pipeline penyebaran dan proses untuk diterapkan ke satu Wilayah pada satu waktu alih-alih menyebarkan ke beberapa Wilayah secara bersamaan. Ini mengurangi kemungkinan kegagalan yang berkorelasi antar Daerah. Untuk mempelajari teknik yang digunakan Amazon untuk mengotomatiskan penerapan perangkat lunak, lihat artikel AWS Builders' Library Mengotomatiskan penerapan yang aman dan hands-off.

4.c: Observabilitas

Ketika Anda merancang untuk Multi-region, pertimbangkan bagaimana Anda akan memantau kesehatan semua komponen di setiap Wilayah untuk mendapatkan pandangan holistik tentang kesehatan Regional. Ini dapat mencakup metrik pemantauan untuk kelambatan replikasi, yang bukan merupakan pertimbangan untuk beban kerja wilayah Tunggal.

Saat Anda membangun arsitektur Multi-wilayah, pertimbangkan untuk mengamati kinerja beban kerja dari Wilayah siaga juga. Ini termasuk pemeriksaan kesehatan dan kenari (pengujian sintetis) yang berjalan dari Wilayah siaga untuk memberikan pandangan luar tentang kesehatan Wilayah primer. Selain itu, Anda dapat menggunakan Amazon CloudWatch Internet Monitor untuk memahami keadaan jaringan eksternal dan kinerja beban kerja Anda dari sudut pandang pengguna akhir. Wilayah utama harus memiliki observabilitas yang sama untuk memantau Wilayah siaga.

Burung kenari dari Wilayah siaga harus memantau metrik pengalaman pelanggan untuk menentukan kesehatan beban kerja secara keseluruhan. Ini diperlukan karena jika ada masalah di Wilayah primer, observabilitas di primer dapat terganggu dan akan memengaruhi kemampuan Anda untuk menilai kesehatan beban kerja.

Dalam hal ini, mengamati di luar Wilayah itu dapat memberikan wawasan. Metrik ini harus digulung menjadi dasbor yang tersedia di setiap Wilayah dan alarm yang dibuat di setiap Wilayah. Karena CloudWatchmerupakan layanan Regional, memiliki alarm di kedua Wilayah merupakan persyaratan. Data pemantauan ini akan digunakan untuk membuat panggilan untuk gagal dari primer ke Region siaga.

4.d: Proses dan prosedur

Waktu terbaik untuk menjawab pertanyaan, “Kapan saya harus gagal?” Jauh sebelum Anda membutuhkannya. Tentukan rencana pemulihan yang mencakup orang, proses, dan teknologi jauh

sebelum suatu masalah, dan uji secara teratur. Tentukan kerangka keputusan pemulihan. Jika ada proses pemulihan yang dipraktikkan dengan baik dan waktu untuk pemulihan dipahami dengan baik, Anda dapat memilih untuk memulai proses pemulihan dengan menggunakan failover yang memenuhi target RTO. Titik waktu ini bisa segera setelah masalah dengan aplikasi di Wilayah utama diidentifikasi, atau bisa lebih jauh menjadi peristiwa ketika opsi pemulihan dalam aplikasi di Wilayah telah habis.

Tindakan failover itu sendiri harus 100 persen otomatis, tetapi keputusan untuk mengaktifkan failover harus dibuat oleh manusia — biasanya sejumlah kecil individu yang telah ditentukan dalam organisasi. Orang-orang ini harus mempertimbangkan kehilangan data dan informasi tentang acara tersebut. Juga, kriteria untuk failover perlu didefinisikan dengan jelas dan dipahami secara global dalam organisasi. Untuk menentukan dan menyelesaikan proses ini, Anda dapat menggunakan sakelar Wilayah Amazon Application Recovery Controller (ARC), yang memungkinkan end-to-end otomatisasi lengkap dan memastikan konsistensi proses yang berjalan selama pengujian dan failover.

Saat Anda membuat paket peralihan Wilayah, paket ini secara otomatis mereplikasi paket Anda di Wilayah utama dan siaga untuk memastikan bahwa tidak ada ketergantungan pada satu Wilayah. Ketika otomatisasi ini ada, tentukan dan ikuti irama pengujian reguler. Ini memastikan bahwa ketika ada peristiwa aktual, respons mengikuti proses yang terdefinisi dengan baik dan dipraktikkan yang dipercaya oleh organisasi. Penting juga untuk mempertimbangkan toleransi yang ditetapkan untuk proses rekonsiliasi data. Konfirmasikan bahwa proses yang diusulkan memenuhi RPO/RTO persyaratan yang ditetapkan.

4.e: Pengujian

Memiliki pendekatan pemulihan yang belum teruji sama dengan tidak memiliki pendekatan pemulihan. Tingkat pengujian dasar adalah menjalankan prosedur pemulihan untuk mengganti Wilayah operasi untuk aplikasi Anda. Terkadang ini disebut sebagai pendekatan rotasi aplikasi. Kami menyarankan Anda membangun kemampuan untuk mengubah Wilayah ke postur operasi normal Anda; Namun, tes ini saja tidak cukup.

Pengujian ketahanan juga penting untuk memvalidasi pendekatan pemulihan aplikasi. Ini melibatkan menyuntikkan skenario kegagalan tertentu, memahami bagaimana aplikasi dan proses pemulihan Anda bereaksi, dan kemudian menerapkan mitigasi apa pun yang diperlukan jika pengujian tidak berjalan sesuai rencana. Menguji prosedur pemulihan Anda tanpa adanya kesalahan tidak akan memberi tahu Anda bagaimana aplikasi Anda berperilaku secara keseluruhan ketika kesalahan terjadi. Anda harus mengembangkan rencana untuk menguji pemulihan Anda terhadap skenario

kegagalan yang diharapkan. AWS Fault Injection Servicemenyediakan daftar skenario yang terus bertambah untuk membantu Anda memulai.

Ini sangat penting untuk aplikasi ketersediaan tinggi, di mana pengujian yang ketat diperlukan untuk memastikan bahwa target kelangsungan bisnis terpenuhi. Menguji kemampuan pemulihan secara proaktif mengurangi risiko kegagalan dalam produksi, yang membangun keyakinan bahwa aplikasi dapat mencapai waktu pemulihan terbatas yang diinginkan. Pengujian rutin juga membangun keahlian operasional, yang memungkinkan tim pulih dengan cepat dan andal dari pemadaman saat terjadi. Melatih elemen manusia, atau proses, dari pendekatan pemulihan Anda sama pentingnya dengan aspek teknis.

4.f: Biaya dan kompleksitas

Implikasi biaya dari arsitektur Multi-region didorong oleh penggunaan infrastruktur yang lebih tinggi, overhead operasional, dan waktu sumber daya. Seperti disebutkan sebelumnya, biaya infrastruktur di Wilayah siaga mirip dengan biaya infrastruktur di Wilayah primer saat pra-penyediaan, sehingga menggandakan total biaya Anda. Kapasitas penyediaan sehingga cukup untuk operasi sehari-hari tetapi masih memiliki kapasitas penyangga yang cukup untuk mentolerir lonjakan permintaan. Kemudian konfigurasikan batas yang sama di setiap Wilayah.

Selain itu, jika Anda mengadopsi arsitektur aktif-aktif, Anda mungkin harus membuat perubahan tingkat aplikasi untuk menjalankan aplikasi Anda dengan sukses dalam arsitektur Multi-region. Perubahan ini dapat memakan waktu dan intensif sumber daya untuk merancang dan mengoperasikan. Minimal, organisasi perlu meluangkan waktu untuk memahami ketergantungan teknis dan bisnis di setiap Wilayah, dan merancang proses failover dan failback.

Tim juga harus melalui latihan failover dan failback normal untuk merasa nyaman dengan runbook yang akan digunakan selama acara. Meskipun latihan ini sangat penting untuk mendapatkan hasil yang diharapkan dari investasi Multi-wilayah, mereka mewakili biaya peluang, dan mengambil waktu dan sumber daya dari kegiatan lain.

4.g: Strategi failover Multi-wilayah Organisasi

Wilayah AWS memberikan batas isolasi kesalahan yang mencegah kegagalan berkorelasi dan menahan dampak dari Layanan AWS gangguan, ketika terjadi, ke satu Wilayah. Anda dapat menggunakan batas kesalahan ini untuk membangun aplikasi Multi-wilayah yang terdiri dari replika independen yang terisolasi kesalahan di setiap Wilayah untuk membatasi skenario nasib bersama. Ini memungkinkan Anda untuk membangun aplikasi Multi-wilayah dan menggunakan

berbagai pendekatan—mulai dari pencadangan dan pemulihan, lampu pilot, hingga aktif-aktif—untuk mengimplementasikan arsitektur Multi-wilayah Anda. Namun, aplikasi biasanya tidak beroperasi secara terpisah, jadi pertimbangkan komponen yang akan Anda gunakan dan dependensinya sebagai bagian dari strategi failover Anda. Umumnya, beberapa aplikasi bekerja sama untuk mendukung cerita pengguna, yang merupakan kemampuan khusus yang ditawarkan kepada pengguna akhir, seperti memposting gambar dan keterangan di aplikasi media sosial atau memeriksa di situs e-commerce. Karena itu, Anda harus mengembangkan strategi failover multi-wilayah organisasi yang menyediakan koordinasi dan konsistensi yang diperlukan untuk membuat pendekatan Anda berhasil.

Ada empat strategi tingkat tinggi yang dapat dipilih organisasi untuk memandu pendekatan Multi-wilayah. Ini terdaftar dari yang paling terperinci hingga pendekatan terluas:

- Failover tingkat komponen
- Failover aplikasi individual
- Failover grafik ketergantungan
- Seluruh failover portofolio aplikasi

Setiap strategi memiliki trade-off dan mengatasi tantangan yang berbeda, termasuk fleksibilitas pengambilan keputusan failover, kemampuan untuk menguji kombinasi failover, keberadaan perilaku modal, dan investasi organisasi dalam perencanaan dan implementasi. Untuk menyelami setiap strategi secara lebih rinci, lihat posting AWS blog Membuat strategi failover Multi-wilayah organisasi.

Panduan utama

- Tinjau semua Layanan AWS kuota untuk memastikan bahwa kuota tersebut berada dalam paritas di semua Wilayah di mana beban kerja akan beroperasi.
- Proses penyebaran harus menargetkan satu Wilayah pada satu waktu alih-alih melibatkan beberapa Wilayah secara bersamaan.
- Metrik tambahan seperti replikasi lag khusus untuk skenario Multi-wilayah dan harus dipantau.
- Memperluas pemantauan untuk beban kerja di luar Wilayah utama. Pantau metrik pengalaman pelanggan untuk setiap Wilayah, dan ukur data ini dari luar setiap Wilayah tempat beban kerja berjalan.
- Uji failover dan failback secara teratur. Terapkan satu runbook untuk proses failover dan failback dan gunakan untuk pengujian dan acara langsung. Runbook untuk pengujian dan acara langsung tidak boleh berbeda.

- Memahami trade-off dari strategi failover. Menerapkan grafik ketergantungan atau seluruh strategi portofolio aplikasi.

Kesimpulan dan sumber daya

Panduan ini mencakup kasus penggunaan umum untuk arsitektur Multi-wilayah, dasar-dasar penerapan arsitektur ini, dan implikasi dari pendekatan ini. Anda dapat menerapkan dasar-dasar ini ke beban kerja apa pun dan menggunakan informasi sebagai kerangka kerja untuk membantu memutuskan apakah arsitektur Multi-wilayah adalah pendekatan yang tepat untuk bisnis Anda.

Untuk informasi selengkapnya, lihat sumber daya berikut:

- AWS Pusat Arsitektur
- AWS Kerangka Well-Architected
- AWS Well-Architected Tool
- Membuat strategi failover Multi-wilayah organisasi (AWS posting blog)
- AWS Kemampuan Multi-Wilayah (AWS Re: artikel posting)

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
Pembaruan untuk sakelar Wilayah ARC	Di bagian 3.c dan 4.d, menambahkan informasi tentang sakelar Wilayah Amazon Application Recovery Controller (ARC) untuk menangani tugas pemulihan.	September 29, 2025
Pembaruan	Pembaruan di seluruh panduan.	Desember 27, 2024
Publikasi awal	—	Desember 20, 2022

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: Migrasikan 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 memproses atau memodifikasi data, seperti menganonimkan, menyunting, atau membuat nama samaran.

Zona Ketersediaan

Lokasi berbeda di dalam AWS Region 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 Region, 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 dapat diakses oleh pengguna akhir. 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, AWS Region, 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 bidikan nol.

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?

interpretabilitas

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. 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 kebijakan berbasis sumber daya), atau menentukan izin maksimum untuk semua akun di organisasi (lihat kebijakan kontrol layanan). AWS Organizations

ketekunan 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 pengoptimalan 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 yang dihosting 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 AWS Region 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). Tipe 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 merujuk 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.

SCADA

Lihat kontrol pengawasan dan akuisisi data.

SCP

Lihat kebijakan kontrol layanan.

Rahasia

Dalam AWS Secrets Manager, informasi rahasia atau terbatas, seperti kata sandi atau kredensil 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 AWS sumber daya Anda.

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. Untuk informasi lebih lanjut, lihat panduan Mengukur ketidakpastian dalam sistem pembelajaran mendalam.

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.