Buku pedoman portofolio untuk migrasi AWS besar

AWS Bimbingan Preskriptif

AWS Bimbingan Preskriptif: Buku pedoman portofolio untuk migrasi AWS besar

Table of Contents

Buku pedoman portofolio untuk migrasi AWS besar

Amazon Web Services (kontributor)

Juli 2024 (sejarah dokumen)

> **Note**
>
> Melakukan penemuan dan penilaian awal tingkat tinggi terhadap portofolio aplikasi merupakan prasyarat untuk menyelesaikan tugas dalam buku pedoman ini. Untuk informasi selengkapnya tentang menyelesaikan proses ini, lihat Panduan penilaian portofolio aplikasi untuk AWS Cloud migrasi.

Dalam migrasi besar, alur kerja portofolio merencanakan gelombang aplikasi untuk migrasi, dan alur kerja migrasi berfokus pada migrasi gelombang tersebut. Saat merencanakan gelombang, alur kerja portofolio bertanggung jawab untuk menilai portofolio, mengumpulkan metadata yang diperlukan untuk migrasi, memprioritaskan aplikasi, dan kemudian menetapkan aplikasi ke gelombang. Gelombang harus berukuran dan dijadwalkan sesuai dengan kapasitas alur kerja migrasi dan harus memperhitungkan kompleksitas aplikasi, dependensi, dan faktor bisnis apa pun, seperti anggaran, tujuan kinerja, ketersediaan sumber daya, dan tenggat waktu. Untuk informasi selengkapnya tentang alur kerja inti dan pendukung, lihat Aliran kerja dalam migrasi besar di buku pedoman Foundation untuk migrasi besar. AWS

Buku pedoman ini memberikan step-by-step pendekatan untuk melakukan penilaian portofolio terperinci untuk proyek migrasi besar, termasuk penilaian aplikasi dan perencanaan gelombang. Ini menjelaskan tugas-tugas alur kerja portofolio, yang mencakup kedua tahap migrasi besar, inisialisasi dan implementasi:

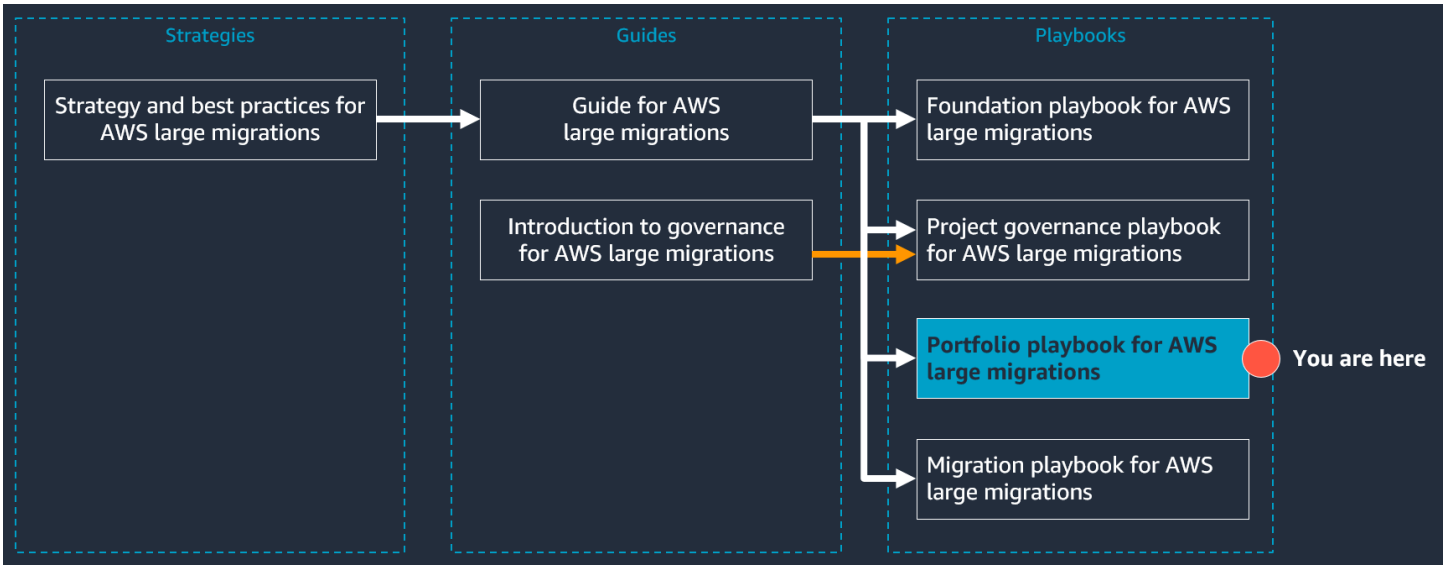
- Pada tahap 1, inisialisasi, Anda memvalidasi penemuan portofolio awal dan strategi migrasi, dan Anda membuat runbook yang menentukan proses dan aturan yang digunakan untuk penilaian portofolio dan perencanaan gelombang. Pada akhir tahap 1, Anda memiliki runbook portofolio dan alat pelacak yang disesuaikan untuk portofolio, proses, dan infrastruktur Anda sendiri.
- Pada tahap 2, implementasikan, Anda menggunakan runbook yang Anda buat di tahap sebelumnya untuk menyelesaikan penilaian portofolio dan rencana gelombang.

Penilaian portofolio terperinci dan perencanaan gelombang bukanlah tugas satu kali. Ini adalah aliran kerja berkelanjutan yang mendukung migrasi. Di pabrik migrasi, penilaian portofolio dan perencanaan gelombang menyediakan bahan baku (server) ke pabrik, jadi Anda harus melanjutkan kegiatan ini hingga proyek migrasi selesai. Untuk informasi selengkapnya tentang model pabrik migrasi, lihat Panduan untuk migrasi AWS besar.

Panduan untuk migrasi besar

Migrasi 300 atau lebih server dianggap sebagai migrasi besar. Tantangan orang, proses, dan teknologi dari proyek migrasi besar biasanya baru bagi sebagian besar perusahaan. Dokumen ini adalah bagian dari seri Panduan AWS Preskriptif tentang migrasi besar ke. AWS Cloud Seri ini dirancang untuk membantu Anda menerapkan strategi dan praktik terbaik yang benar sejak awal, untuk merampingkan perjalanan Anda ke cloud.

Gambar berikut menunjukkan dokumen lain dalam seri ini. Tinjau strategi terlebih dahulu, lalu panduannya, lalu lanjutkan ke buku pedoman. Untuk mengakses seri lengkap, lihat Migrasi besar ke file. AWS Cloud



Tentang runbook, alat, dan templat

Di buku pedoman ini, Anda membuat runbook berikut:

- Runbook prioritas aplikasi
- Runbook manajemen metadata
- Runbook perencanaan gelombang

Selain itu, Anda membuat alat berikut, yang Anda gunakan untuk melacak kemajuan atau mendokumentasikan keputusan dan informasi penting lainnya:

- Lembar skor kompleksitas aplikasi
- Lembar kerja status target aplikasi
- Pelacak kemajuan penilaian portofolio
- Kuesioner untuk pemilik aplikasi
- Dasbor perencanaan dan migrasi gelombang

Sebaiknya gunakan templat buku pedoman portofolio dan kemudian menyesuaikannya untuk portofolio, proses, dan lingkungan Anda. Petunjuk dalam buku pedoman ini memberi tahu Anda kapan dan bagaimana menyesuaikan masing-masing templat ini. Buku pedoman ini mencakup templat berikut:

- Lembar kerja status target aplikasi - Anda menggunakan template ini untuk menentukan status masa depan aplikasi di AWS Cloud ketika aplikasi atau strategi migrasi sangat kompleks.
- Template dasbor untuk perencanaan dan migrasi gelombang — Anda menggunakan template ini untuk menyusun metadata penting, menganalisis portofolio aplikasi, mengidentifikasi dependensi, dan merencanakan gelombang migrasi.
- Template pelacakan kemajuan untuk penilaian portofolio - Anda menggunakan template ini untuk melacak kemajuan setiap aplikasi melalui alur kerja portofolio.
- Template kuesioner untuk pemilik aplikasi — Anda menggunakan template ini dalam proses penyelaman mendalam aplikasi untuk mengumpulkan informasi tentang aplikasi langsung dari pemilik aplikasi.
- Template Runbook untuk prioritas aplikasi — Template ini adalah titik awal untuk membangun prioritas aplikasi Anda sendiri dan proses penyelaman mendalam.
- Template Runbook untuk manajemen metadata - Template ini adalah titik awal untuk membangun identifikasi metadata dan proses pengumpulan Anda sendiri.
- Template Runbook untuk perencanaan gelombang - Template ini adalah titik awal untuk membangun proses perencanaan gelombang Anda sendiri.
- Templat lembar skor untuk kompleksitas aplikasi — Anda dapat menggunakan template ini untuk mengevaluasi kompleksitas migrasi setiap aplikasi ke cloud, dan kemudian Anda dapat menggunakan skor yang dihasilkan selama proses prioritas aplikasi.

Tahap 1: Menginisialisasi migrasi besar

Pada tahap inisialisasi, Anda menentukan runbook yang Anda gunakan untuk menyelesaikan penilaian portofolio terperinci dan rencana gelombang dalam tahap implementasi. Jika anggota tim lain bertanggung jawab untuk menentukan runbook dalam proyek migrasi besar Anda, lewati ke Tahap 2: Menerapkan migrasi besar, di mana Anda akan menggunakan runbook untuk memigrasikan gelombang aplikasi dan server. Dengan mendokumentasikan keputusan yang dibuat pada tahap ini, Anda membuat runbook yang dapat ditindaklanjuti. Misalnya, membuat keputusan tentang pertanyaan-pertanyaan berikut mengarah ke prosedur standar yang Anda dokumentasikan dalam runbook portofolio Anda:

- Metadata migrasi apa yang diperlukan, dan bagaimana Anda mengumpulkannya?
- Bagaimana Anda memprioritaskan aplikasi dan melakukan penyelaman mendalam?
- Bagaimana Anda merencanakan gelombang?

Pada tahap 1, Anda menghabiskan banyak waktu untuk menentukan aturan dan membangun runbook karena aktivitas di runbook diulang berkali-kali di tahap 2 untuk mendukung migrasi.

Tahap 1 terdiri dari tugas dan langkah berikut

- Tugas 1: Melakukan penemuan awal dan memvalidasi strategi migrasi
  - Langkah 1: Validasi data penemuan
  - Langkah 2: Identifikasi driver bisnis dan teknis
  - Langkah 3: Validasi strategi migrasi
  - Langkah 4: Validasi pola migrasi
- Tugas 2: Mendefinisikan proses untuk mengidentifikasi, mengumpulkan, dan menyimpan metadata
  - Langkah 1: Tentukan metadata yang diperlukan
  - Langkah 2: Membangun penyimpanan metadata dan proses pengumpulan
  - Langkah 3: Dokumentasikan persyaratan metadata dan proses pengumpulan di runbook
- Tugas 3: Mendefinisikan proses prioritas aplikasi
  - Langkah 1: Tentukan proses prioritas aplikasi
  - Langkah 2: Tentukan aturan prioritas aplikasi
  - Langkah 3: Selesaikan proses prioritas aplikasi
- Tugas 4: Mendefinisikan proses penyelaman mendalam aplikasi

Tugas 1: Melakukan penemuan awal dan memvalidasi strategi migrasi

Langkah pertama penilaian portofolio dalam proyek migrasi besar adalah memahami informasi yang Anda miliki saat ini, driver bisnis dan teknis, dan keputusan strategi migrasi apa pun yang telah dibuat. Hasil penilaian portofolio adalah untuk terus memasukkan metadata migrasi, rencana gelombang, dan strategi migrasi ke dalam alur kerja migrasi. Berdasarkan informasi yang dikumpulkan, Anda menganalisis kesenjangan dan memutuskan langkah selanjutnya. Anda dapat melewati beberapa bagian dalam buku pedoman ini jika Anda telah menyelesaikan analisis dan tugas. Tugas ini terdiri dari langkah-langkah berikut:

- Langkah 1: Validasi data penemuan
- Langkah 2: Identifikasi driver bisnis dan teknis
- Langkah 3: Validasi strategi migrasi
- Langkah 4: Validasi pola migrasi

Langkah 1: Validasi data penemuan

Pada fase mobilisasi, Anda mungkin telah menyelesaikan penilaian portofolio awal, dan jika demikian, Anda dapat menggunakan kembali data penemuan tersebut dalam fase migrasi. Jika tidak, jangan khawatir. Buku pedoman ini akan memandu Anda melalui apa yang diperlukan untuk mendukung migrasi besar Anda.

Migrasi besar biasanya memiliki banyak data. Misalnya, Anda memiliki:

- Metadata tentang server sumber, aplikasi, dan database
- Informasi tentang portofolio TI Anda dari database manajemen konfigurasi (CMDB)
- Data dari alat penemuan yang membantu Anda lebih memahami keadaan dan dependensi saat ini
- Metadata untuk sumber daya target AWS

Tentang jenis metadata

Berikut ini adalah tiga jenis utama metadata yang diperlukan untuk mendukung migrasi besar:

- Metadata portofolio sumber — Metadata portofolio sumber adalah metadata tentang server sumber, aplikasi, dan database Anda. Anda bisa mendapatkan metadata dari CMDB yang ada, alat penemuan, atau bahkan dari pemilik aplikasi. Anda dapat menemukan daftar lengkap jenis metadata ini di sini, dan berikut ini adalah beberapa contoh:
  - Nama Server
  - Alamat IP server
  - Sistem operasi server (OS)
  - Penyimpanan server, CPU, memori, dan input/output operasi per detik (IOPS)
  - Nama aplikasi
  - Pemilik aplikasi
  - Application-to-application dependensi
  - Unit bisnis
  - Application-to-server pemetaan
  - Application-to-database pemetaan
  - Jenis dan ukuran database
  - Jenis dan ukuran penyimpanan
  - Metadata dependensi
  - Data kinerja dan penggunaan
- Metadata lingkungan target — Ini adalah jenis metadata yang membantu Anda memigrasikan server ke lingkungan target. Anda perlu membuat keputusan tentang lingkungan target. Anda bisa mendapatkan beberapa metadata ini dari alat penemuan. Berikut ini adalah beberapa contoh jenis metadata ini:
  - Subnet target
  - Target kelompok keamanan

  - Jenis instans target
  - Peran Target AWS Identity and Access Management (IAM)
  - Alamat IP target
  - ID AWS akun target
  - AWS Wilayah Target
  - AWS Layanan target
  - Desain arsitektur aplikasi target
- Metadata perencanaan gelombang — Metadata perencanaan gelombang adalah jenis metadata yang membantu Anda mengelola migrasi. Berikut ini adalah contoh jenis metadata ini:
  - ID Gelombang
  - Waktu mulai gelombang
  - Waktu pemotongan gelombang
  - Pemilik gelombang
  - Pemetaan gelombang ke application/server/database/move grup

Validasi data penemuan Anda

Penting untuk memahami data penemuan Anda saat ini sebelum membuat keputusan apa pun. Anda mungkin tidak memiliki semua informasi pada tahap migrasi ini. Buku pedoman ini membantu Anda menentukan persyaratan metadata dan membantu Anda mengumpulkan metadata secara efisien. Tanyakan pada diri Anda pertanyaan-pertanyaan berikut untuk mengidentifikasi metadata apa yang saat ini tersedia dan di mana lokasinya:

- Sudahkah Anda menggunakan alat apa pun untuk melakukan penilaian migrasi, seperti Migration Evaluator?
- Sudahkah Anda menerapkan alat penemuan apa pun di lingkungan Anda, seperti Migrasi dan Modernisasi Cloud Flexera One?
- Apakah Anda memiliki CMDB yang memiliki up-to-date informasi paling banyak untuk portofolio TI Anda?
- Sudahkah Anda menyelesaikan penilaian portofolio awal dalam fase mobilisasi?
- Sudahkah Anda menyelesaikan perencanaan gelombang awal?
- Sudahkah Anda menyelesaikan desain lingkungan target awal?
- Apa sumber dari setiap jenis metadata?

- Apakah Anda memiliki akses ke semua metadata?
- Bagaimana Anda mengakses semua metadata?
- Sudahkah Anda mendokumentasikan proses mengakses metadata?

Langkah 2: Identifikasi driver bisnis dan teknis

Penggerak bisnis dan teknologi sangat penting ketika mempertimbangkan strategi dan pola migrasi tingkat tinggi untuk setiap aplikasi. Anda harus memahami driver yang unik untuk migrasi Anda. Anda menggunakan driver bisnis dan teknis ini saat memvalidasi strategi migrasi dan menentukan aturan pemetaan aplikasi.

Pengemudi bisnis umum

Penggerak bisnis adalah faktor yang terkait dengan tujuan atau batasan bisnis yang harus Anda pertimbangkan ketika merencanakan migrasi besar, seperti kontrak yang kedaluwarsa, pertumbuhan yang cepat, atau anggaran. Berikut ini adalah driver bisnis yang umum:

- Keluar dari pusat data — Anda harus bermigrasi secepat mungkin ke cloud. Misalnya, kontrak pusat data akan segera kedaluwarsa.
- Mengurangi biaya operasional dan risiko — Anda ingin mengurangi biaya atau risiko yang terkait dengan pengoperasian lingkungan lokal.
- Fleksibilitas — Anda perlu pindah ke cloud sebagai arah strategis untuk mempersiapkan perubahan di masa depan bisnis.
- Menumbuhkan bisnis — Anda harus dapat dengan cepat mempercepat pengembangan dan inovasi atau mengakomodasi pertumbuhan yang cepat.
- Menggunakan data secara cerdas — Anda ingin memanfaatkan kecerdasan buatan berbasis cloud, pembelajaran mesin, dan Internet of Things (IoT) yang dapat meramalkan pertumbuhan perusahaan Anda dan memberikan wawasan tentang perilaku pelanggan.
- Meningkatkan keamanan dan kepatuhan - Anda perlu memanfaatkan program kepatuhan yang sudah dibangun ke dalam infrastruktur AWS Cloud, atau Anda ingin memanfaatkan alat keamanan berbasis perangkat lunak yang dapat memperingatkan Anda tentang kemungkinan ancaman terhadap data Anda.
- Ketersediaan sumber daya — Memiliki sumber daya yang terbatas atau pengalaman internal yang terbatas dapat mengarahkan Anda untuk memilih strategi yang memindahkan aplikasi tanpa modifikasi.

Driver teknis umum

Driver teknis adalah faktor yang terkait dengan tujuan teknis atau batasan yang harus Anda pertimbangkan ketika merencanakan migrasi besar, seperti arsitektur saat ini. Berikut ini adalah driver teknis umum:

- Perangkat keras atau perangkat lunak end-of-support — Perangkat keras atau perangkat lunak Anda mendekati akhir siklus hidupnya, dan Anda perlu menyegarkannya karena vendor tidak lagi mendukungnya.
- Integrasi teknologi — Anda mendapatkan akses ke infrastruktur global yang memungkinkan Anda menskalakan aplikasi Anda dengan cepat dan strategis. Anda dapat mendunia dengan cepat dengan layanan dan infrastruktur global yang siap untuk Anda ketuk.
- Batasan penyimpanan dan komputasi — Pusat data Anda tidak memiliki kapasitas untuk lebih banyak penyimpanan atau server, dan Anda perlu mencari tempat lain untuk memperluas.
- Persyaratan skalabilitas dan ketahanan — Aplikasi Anda telah mengalami downtime di masa lalu, dan Anda ingin menggunakan cloud untuk meningkatkan tujuan titik pemulihan (RPO) dan tujuan waktu pemulihan (RTO).
- Modernisasi arsitektur aplikasi — Anda ingin memanfaatkan cloud dan mengubah aplikasi Anda menjadi cloud-native.
- Meningkatkan kinerja — Performa aplikasi Anda buruk selama musim puncak, Anda ingin meningkatkan dan menurunkan skala secara otomatis agar sesuai dengan permintaan.

Perbarui runbook

1. Dalam template playbook portofolio, buka template Runbook untuk prioritas aplikasi (format Microsoft Word).
2. Di bagian Driver bisnis dan teknis, catat driver yang Anda identifikasi untuk proyek migrasi besar Anda.
3. Simpan runbook prioritas aplikasi Anda.

Langkah 3: Validasi strategi migrasi

Memilih strategi migrasi sangat penting untuk migrasi besar. Anda harus memverifikasi bahwa strategi migrasi yang Anda pilih memenuhi harapan, batasan, dan persyaratan organisasi. Untuk

informasi selengkapnya tentang strategi migrasi yang tersedia, lihat Panduan untuk migrasi AWS besar.

Anda mungkin telah memilih strategi migrasi dalam fase mobilisasi atau selama penilaian portofolio awal. Pada langkah ini, Anda menggunakan driver bisnis dan teknis untuk memilih dan memvalidasi strategi migrasi untuk portofolio Anda.

Strategi migrasi Anda mungkin berubah saat Anda terus menilai portofolio dan memulai migrasi. Pada tahap ini, tujuannya adalah untuk memahami distribusi umum portofolio Anda ke setiap strategi migrasi. Memilih strategi migrasi sangat penting untuk langkah berikutnya, memvalidasi pola migrasi terperinci.

Pilih dan validasi strategi migrasi

Evaluasi portofolio dan pilih strategi migrasi sebagai berikut:

1. Tinjau semua driver teknis dan bisnis yang Anda identifikasi pada langkah sebelumnya, dan prioritaskan driver berdasarkan kebutuhan bisnis Anda.
2. Petakan setiap driver bisnis dan teknis ke strategi migrasi. Tabel berikut adalah contohnya.

Prioritas	Pengemudi bisnis atau teknis	Strategi migrasi
1	Keluar dari pusat data pada tanggal yang ditentukan	Rehost aplikasi sebanyak mungkin, dan replatform dan refactor hanya jika rehost tidak memungkinkan.
2	Mengurangi biaya operasional dan risiko	Untuk mempercepat migrasi, rehost aplikasi sebanyak mungkin.
3	Perangkat keras atau perangkat lunak end-of-support	Rehost aplikasi yang didukung dan aplikasi replatform yang tidak didukung ke perangkat keras dan perangkat lunak yang lebih baru di cloud.

Prioritas	Pengemudi bisnis atau teknis	Strategi migrasi
4	Ketersediaan sumber daya	Rehost ke AWS Managed Services (AMS) untuk mengurangi overhead operasional.

3. Dengan menimbang setiap driver bisnis dan teknis serta mengevaluasi portofolio Anda pada tingkat tinggi, perkirakan bagaimana aplikasi harus didistribusikan di antara setiap strategi migrasi. Adalah umum untuk melihat konflik antara pengemudi. Pemangku kepentingan proyek perlu bekerja sama dan membuat keputusan akhir untuk menyelesaikan konflik. Berikut ini adalah contoh bagaimana Anda dapat mendistribusikan portofolio Anda ke setiap strategi migrasi:
   - Rehost - 60%
   - Replatform — 15%
   - Pensiun — 10%
   - Pertahankan - 5%
   - Pembelian kembali — 5%
   - Refactor - 5%

Jangan melanjutkan migrasi hingga Anda memilih strategi migrasi tingkat tinggi untuk portofolio Anda.

Perbarui runbook

1. Buka runbook prioritas aplikasi Anda.
2. Di bagian Strategi migrasi, catat bagaimana beban kerja aplikasi didistribusikan di antara tujuh strategi migrasi. Contoh:
   - Rehost - 60%
   - Replatform — 15%
   - Pensiun — 10%
   - Pertahankan - 5%
   - Pembelian kembali — 5%
   - Refactor - 5%
3. Simpan runbook prioritas aplikasi Anda.

Langkah 4: Validasi pola migrasi

Tentang pola migrasi

Pola migrasi adalah tugas migrasi berulang yang merinci strategi migrasi, tujuan migrasi, dan aplikasi atau layanan migrasi yang digunakan. Contohnya adalah Rehost ke Amazon Elastic Compute Cloud (Amazon EC2) menggunakan. AWS Application Migration Service AWS Layanan dan solusi berikut sering direferensikan dalam pola migrasi umum:

- AWS App2Container
- AWS Application Migration Service (AWS MGN)
- AWS CloudFormation
- AWS Database Migration Service (AWS DMS)
- AWS DataSync
- Amazon Elastic Compute Cloud (Amazon EC2)
- Amazon Elastic Container Service (Amazon ECS)
- Amazon Elastic File System (Amazon EFS)
- AWS Solusi Pabrik Migrasi Cloud
- Amazon Relational Database Service (Amazon RDS)
- AWS Schema Conversion Tool (AWS SCT)
- AWS Transfer Family

Mirip dengan memilih strategi migrasi, Anda mungkin telah mengidentifikasi pola migrasi Anda di fase sebelumnya. Namun, Anda harus memvalidasi mereka dan memastikan pola telah ditentukan dan didokumentasikan. Tabel berikut mencantumkan strategi dan pola migrasi umum.

ID	Strategi	Pola
1	Rehost	Rehost ke Amazon EC2 menggunakan Layanan Migrasi Aplikasi atau Pabrik Migrasi Cloud

ID	Strategi	Pola
2	Platform Ulang	Replatform ke Amazon RDS menggunakan dan AWS DMS AWS SCT
3	Platform Ulang	Replatform ke Amazon EC2 menggunakan CloudForm ation **Note** CloudFormation template membangun infrastruktur baru di AWS Cloud.
4	Platform Ulang	Platform ulang ke Amazon EFS menggunakan AWS DataSync atau AWS Transfer Family
5	Platform Ulang	Replatform ke Amazon ECS menggunakan App2Container AWS
6	Platform Ulang	Replatform mainframe atau server midrange ke Amazon EC2 menggunakan emulator
7	Platform Ulang	Replatform dari Windows ke Linux di Amazon EC2
8	Pensiun	Pensiun aplikasi
9	Mempertahankan	Pertahankan di tempat

ID	Strategi	Pola
10	Pembelian kembali	Beli kembali dan tingkatkan ke SaaS
11	Refactor atau arsitek ulang	Arsitek ulang aplikasi

Perbarui runbook

Pada titik ini, Anda menentukan pola di tingkat portofolio. Kemudian di buku pedoman ini, Anda memetakan setiap aplikasi ke pola migrasi yang sesuai.

1. Buka runbook prioritas aplikasi Anda.
2. Di bagian Pola migrasi, catat pola migrasi yang telah Anda identifikasi dan validasi. Tetapkan setiap pola ID unik dan catat strategi migrasi untuk pola tersebut.
3. Simpan runbook prioritas aplikasi Anda.

Perhatikan bahwa pola migrasi mungkin berubah seiring kemajuan Anda. Anda dapat mengubah strategi dan pola migrasi nanti, saat Anda menemukan informasi baru, mengubah cakupan beban kerja, atau bahkan memutuskan untuk menggunakan AWS layanan baru.

Kriteria keluar tugas

Jika Anda belum mengidentifikasi strategi dan pola migrasi Anda dari perspektif portofolio tingkat tinggi, kami sangat menyarankan Anda bekerja dengan tim teknis untuk mendefinisikannya sebelum melanjutkan ke tugas berikutnya. Penilaian portofolio dan perencanaan gelombang tergantung pada pemahaman strategi dan pola migrasi. Anda tidak perlu memiliki daftar lengkap pola migrasi sebelum melanjutkan. Anda dapat menambahkan pola baru dan menyesuaikan strategi Anda saat Anda pergi.

Lanjutkan ke tugas berikutnya setelah Anda menyelesaikan yang berikut:

- Anda memiliki akses ke data penemuan terbaru dan memahaminya.
- Anda telah mengidentifikasi driver bisnis dan teknis untuk migrasi Anda.
- Anda telah memilih dan memvalidasi strategi migrasi, berdasarkan driver bisnis dan teknis Anda.
- Anda telah memilih dan memvalidasi pola migrasi.
- Anda telah mendokumentasikan hal-hal berikut dalam runbook prioritas aplikasi Anda:

- Driver bisnis dan teknis
- Strategi migrasi
- Pola migrasi

Tugas 2: Mendefinisikan proses untuk mengidentifikasi, mengumpulkan, dan menyimpan metadata

Di tugas sebelumnya, Anda memvalidasi data penemuan awal, strategi migrasi, dan pola migrasi untuk migrasi besar Anda. Dalam tugas ini, Anda mengidentifikasi metadata apa yang diperlukan dan memutuskan bagaimana Anda akan mengumpulkannya. Tugas ini terdiri dari langkah-langkah berikut:

- Langkah 1: Tentukan metadata yang diperlukan
- Langkah 2: Membangun penyimpanan metadata dan proses pengumpulan
- Langkah 3: Dokumentasikan persyaratan metadata dan proses pengumpulan di runbook

Saat Anda menyelesaikan langkah-langkah di bagian ini, pertimbangkan seluruh siklus migrasi dari perspektif metadata. Pertimbangkan penilaian portofolio, perencanaan gelombang, migrasi, pengujian, aktivitas pasca-pemotongan, dan kemudian analisis semua kemungkinan kasus penggunaan dan kasus penggunaan terkait. Memikirkan informasi yang Anda butuhkan untuk menyelesaikan proses migrasi penuh membantu Anda mengidentifikasi semua metadata untuk pola itu.

Langkah 1: Tentukan metadata yang diperlukan

Sebelum Anda dapat menentukan atribut metadata yang diperlukan, Anda harus memahami pola migrasi. Misalnya, Anda memerlukan metadata yang berbeda untuk memigrasikan server ke Amazon EC2 dan untuk memigrasikan database ke Amazon RDS. Sebagian besar pola terdiri dari banyak tugas kecil. Untuk melakukan pola migrasi, Anda perlu mengetahui atribut metadata apa yang diperlukan dan kemudian mengumpulkan metadata untuk aplikasi tersebut. Anda harus menentukan dan mengumpulkan metadata yang diperlukan dalam tahap inisialisasi sehingga Anda dapat melakukan migrasi secara efisien dan tanpa penundaan dalam tahap implementasi.

Orang atau tim yang mendefinisikan atribut metadata dimulai dengan mendefinisikan langkah dan tugas yang diperlukan untuk melakukan pola migrasi. Tugas menentukan metadata apa yang dibutuhkan, jadi mengerjakan setiap tugas membangun kumpulan metadata yang diperlukan secara

komprehensif. Orang yang menentukan metadata apa yang diperlukan biasanya perlu memiliki pemahaman yang komprehensif tentang bagaimana menyelesaikan pola migrasi. Koordinasi dengan orang yang menulis buku runbook migrasi mungkin diperlukan. Untuk informasi selengkapnya, lihat buku pedoman Migrasi untuk migrasi AWS besar.

Selama migrasi besar, ada banyak proses yang tersebar di semua alur kerja yang memiliki ketergantungan pada metadata. Memiliki metadata yang tepat waktu dan akurat memiliki dampak yang luas dan signifikan terhadap keberhasilan migrasi besar.

Pada langkah ini, Anda menentukan pola atau tugas dan kemudian menggunakan definisi untuk mengidentifikasi metadata yang diperlukan.

Identifikasi komponen kunci dari pola migrasi dan tugas pendukung

Pada langkah ini, untuk setiap pola migrasi atau tugas pendukung, Anda menentukan komponen utama, seperti tindakan, objek sumber, objek target, dan alat yang digunakan. Anda kemudian memberi nama pola atau tugas berdasarkan jawaban Anda.

Tugas pendukung meliputi kegiatan operasional yang perlu dilakukan oleh portofolio dan alur kerja migrasi selama migrasi, seperti perencanaan gelombang, prioritas aplikasi, analisis ketergantungan, tata kelola, pemulihan bencana, pengujian kinerja, atau pengujian penerimaan pengguna. Karena Anda memerlukan metadata untuk mendukung tugas ini, Anda melakukan langkah-langkah ini untuk pola migrasi dan tugas pendukung.

1. Tindakan — Identifikasi strategi migrasi atau tugas pendukung. Ingatlah bahwa satu tindakan mungkin memiliki tindakan lain yang terkait dengannya. Misalnya, Anda mungkin ingin menentukan operasi untuk migrasi. Contoh tindakan meliputi:
   - Strategi migrasi, seperti rehost, replatform, atau relokasi
   - Perencanaan gelombang
   - Prioritas aplikasi dan analisis ketergantungan
   - Operasi
   - Tata kelola
   - Pemulihan bencana
   - Pengujian, seperti pengujian kinerja atau pengujian penerimaan pengguna (UAT)
2. Objek sumber — Identifikasi objek sumber tempat tindakan akan dilakukan. Contoh objek sumber meliputi:
   - Gelombang

- Server
- Basis Data
- Berbagi file
- Aplikasi

3. Alat — Identifikasi layanan atau alat yang digunakan untuk melakukan tindakan. Anda dapat menggunakan lebih dari satu alat atau layanan. Contoh alat meliputi:
   - AWS Application Migration Service
   - AWS DataSync
   - AWS Database Migration Service (AWS DMS)
   - AWS Backup
   - Alat pemantauan kinerja
4. Objek target — Identifikasi objek target, layanan, atau lokasi di mana sumber akan berada ketika tindakan selesai. Contoh objek, layanan, atau lokasi meliputi:
   - Amazon Elastic Compute Cloud (Amazon EC2)
   - Amazon Relational Database Service (Amazon RDS)
   - Amazon Elastic File System (Amazon EFS)
   - Amazon Elastic Container Service (Amazon ECS)
   - Rencana gelombang
5. Nama pola - Gabungkan jawaban Anda ke langkah-langkah sebelumnya sebagai berikut:

   <action><source object> on/to <target object>menggunakan <tool>

   Berikut ini adalah beberapa contohnya:
   - Rehost (aksi) gelombang, aplikasi, atau server (objek sumber) ke Amazon EC2 (objek target) menggunakan Layanan Migrasi Aplikasi atau Pabrik Migrasi Cloud (alat)
   - Pembagian file replatform (tindakan) (objek sumber) ke Amazon EFS (objek target) menggunakan DataSync (alat)
   - Platform ulang basis data (tindakan) (objek sumber) ke Amazon RDS (objek target) menggunakan (alat AWS DMS)
   - Pemantauan kinerja (tindakan) aplikasi (objek sumber) di Amazon EC2 (objek target) menggunakan Amazon CloudWatch (alat)
   - Cadangkan server (tindakan) (objek sumber) di Amazon EC2 (objek target) menggunakan AWS Backup (alat) setelah migrasi

- Perencanaan gelombang (aksi) gelombang, aplikasi, atau server (objek sumber) untuk membuat rencana gelombang (objek target)

Berikut ini adalah contoh cara merekam Pola 1: Rehost ke Amazon EC2 menggunakan Application Migration Service atau Cloud Migration Factory dari tabel pola migrasi.

Pattern ID	1
Pattern name	Rehost ke Amazon EC2 menggunakan Layanan Migrasi Aplikasi atau Pabrik Migrasi Cloud
Action	Migrasi rehost
Source object	Gelombang, aplikasi, atau server
Tools	Layanan Migrasi Aplikasi atau Pabrik Migrasi Cloud
Target object	Amazon EC2

Tentukan metadata yang diperlukan untuk setiap pola atau tugas

Sekarang setelah Anda menentukan pola atau tugas, Anda menentukan metadata yang diperlukan untuk objek sumber, objek target, alat, dan informasi bisnis lainnya. Untuk menjelaskan proses ini, pedoman ini menggunakan Pola 1: Rehost ke Amazon EC2 menggunakan Layanan Migrasi Aplikasi atau Pabrik Migrasi Cloud dari tabel pola migrasi sebagai contoh. Perhatikan bahwa untuk beberapa pola atau tugas, beberapa langkah mungkin tidak berlaku.

1. Menganalisis objek target — Bekerja mundur dari objek target, secara manual membuat objek dan mengidentifikasi metadata yang diperlukan untuk mendukungnya. Tangkap metadata seperti yang ditunjukkan pada tabel berikut.

   Misalnya, saat Anda membuat instans EC2, Anda harus memilih jenis instans, jenis penyimpanan, ukuran penyimpanan, subnet, grup keamanan, dan tag. Tabel berikut mencakup contoh atribut metadata yang mungkin Anda perlukan jika objek target Anda adalah instans EC2.

Nama atribut	Jenis objek	Deskripsi atau tujuan
`target_subnet`	Target EC2 misalnya	Subnet dari instans EC2 target
`target_subnet_test`	Target EC2 misalnya	Uji subnet dari instans EC2 target
`target_security_group`	Target EC2 misalnya	Kelompok keamanan dari instans EC2 target
`target_security_group_test`	Target EC2 misalnya	Uji kelompok keamanan dari instans EC2 target
`IAM_role`	Target EC2 misalnya	AWS Identity and Access Management (IAM) peran instans EC2 target
`instance_type`	Target EC2 misalnya	Jenis instans dari instans EC2 target
`AWS_account_ID`	Target EC2 misalnya	AWS akun untuk meng-host instans EC2 target
`AWS_Region`	Target EC2 misalnya	AWS Wilayah untuk menjadi tuan rumah instans EC2 target

2. Analisis alat - Gunakan alat untuk membuat objek target dan periksa perbedaannya. Tangkap metadata khusus alat seperti yang ditunjukkan dalam tabel berikut, dan hapus atribut dari tabel sebelumnya jika tidak didukung oleh alat migrasi. Misalnya, Anda tidak dapat menyesuaikan jenis OS dan ukuran penyimpanan untuk Layanan Migrasi Aplikasi karena alat migrasi rehost adalah like-for-like. Oleh karena itu, Anda akan menghapus OS target dan ukuran disk target jika atribut ini disertakan dalam tabel sebelumnya. Dalam tabel contoh sebelumnya, semua atribut didukung oleh alat, jadi tidak ada tindakan yang diperlukan.

   Tabel berikut mencakup contoh metadata yang mungkin Anda perlukan untuk alat.

Nama atribut	Jenis objek	Deskripsi atau tujuan
`AWS_account_ID`	Alat (Layanan Migrasi Aplikasi)	AWS ID akun untuk AWS Application Migration Service
`AWS_Region`	Alat (Layanan Migrasi Aplikasi)	AWS Wilayah untuk Layanan Migrasi Aplikasi
`replication_server_subnet`	Alat (Layanan Migrasi Aplikasi)	Subnet untuk server replikasi Layanan Migrasi Aplikasi
`replication_server_security_group`	Alat (Layanan Migrasi Aplikasi)	Grup keamanan untuk server replikasi Layanan Migrasi Aplikasi

3. Menganalisis objek sumber - Tentukan metadata yang diperlukan untuk objek sumber dengan menilai tindakan sebagai berikut:
   - Untuk memigrasi server, Anda perlu mengetahui nama server sumber dan nama domain yang memenuhi syarat (FQDN) untuk terhubung ke server.
   - Untuk memigrasikan aplikasi bersama dengan servernya, Anda perlu mengetahui nama aplikasi, lingkungan aplikasi, dan application-to-server pemetaan.
   - Untuk melakukan penilaian portofolio, memprioritaskan aplikasi, atau menentukan grup pindah, Anda perlu mengetahui pemetaan, application-to-server pemetaan, application-to-database dan dependensi. application-to-application
   - Untuk mengelola gelombang, Anda perlu mengetahui ID gelombang dan waktu mulai dan akhir gelombang.

   Tabel berikut mencakup contoh metadata yang mungkin Anda perlukan untuk objek sumber.

Nama atribut	Jenis objek	Deskripsi atau tujuan
`wave_ID`	Gelombang sumber	ID gelombang (misalnya: gelombang 10)
`wave_start_date`	Gelombang sumber	Tanggal mulai untuk gelombang

Nama atribut	Jenis objek	Deskripsi atau tujuan
`wave_cutover_date`	Gelombang sumber	Tanggal cutover untuk gelombang
`wave_owner`	Gelombang sumber	Pemilik gelombang
`app_name`	Aplikasi sumber	Nama aplikasi sumber
`app_to_server_mapping`	Aplikasi sumber	Application-to-server hubungan
`app_to_DB_mapping`	Aplikasi sumber	Application-to-database hubungan
`app_to_app_dependencies`	Aplikasi sumber	Dependensi eksternal aplikasi
`server_name`	Server sumber	Nama server sumber
`server_FQDN`	Server sumber	Nama domain yang sepenuhnya memenuhi syarat dari server sumber
`server_OS_family`	Server sumber	Keluarga sistem operasi (OS) dari server sumber (misalnya: Windows atau Linux)
`server_OS_version`	Server sumber	Versi OS dari server sumber (misalnya: Windows Server 2003)
`server_environment`	Server sumber	Lingkungan server sumber (misalnya: pengembangan, produksi, atau pengujian)
`server_tier`	Server sumber	Tingkat server sumber (misalnya: web, database, atau aplikasi)

Nama atribut	Jenis objek	Deskripsi atau tujuan
`CPU`	Server sumber	Jumlah CPUs di server sumber
`RAM`	Server sumber	Ukuran RAM dari server sumber
`disk_size`	Server sumber	Ukuran disk dari server sumber

4. Pertimbangkan atribut lain — Selain tindakan utama, pertimbangkan tindakan dan atribut lain yang terkait dengan objek target atau aplikasi. Untuk pola contoh, Pola 1: Rehost ke Amazon EC2 menggunakan Layanan Migrasi Aplikasi atau Pabrik Migrasi Cloud, tindakannya adalah rehost, dan objek targetnya adalah Amazon EC2. Tindakan terkait lainnya untuk objek target ini mungkin termasuk mencadangkan ke Amazon EC2, memantau instans EC2 setelah migrasi, dan menggunakan tag untuk mengelola biaya yang terkait dengan instans EC2. Anda mungkin juga ingin mempertimbangkan atribut aplikasi lain yang membantu Anda mengelola migrasi, seperti pemilik aplikasi, yang mungkin perlu Anda hubungi untuk pertanyaan atau tujuan pemotongan.

   Tabel berikut mencakup contoh metadata tambahan yang umum digunakan. Tabel ini mencakup tag untuk instans EC2 target Anda. Untuk informasi selengkapnya tentang tag dan cara menggunakannya, lihat Menandai sumber daya Amazon EC2 Anda di dokumentasi Amazon EC2.

Nama atribut	Jenis objek	Deskripsi atau tujuan
`Name`	Target EC2 misalnya (tag)	Tag untuk menentukan nama instans EC2 target
`app_owner`	Aplikasi sumber	Pemilik aplikasi sumber
`business_unit`	Target EC2 misalnya (tag)	Tag untuk mengidentifikasi unit bisnis untuk instans EC2 target (misalnya: SDM, keuangan, atau TI)

Nama atribut	Jenis objek	Deskripsi atau tujuan
`cost_center`	Target EC2 misalnya (tag)	Tag untuk mengidentifikasi pusat biaya untuk instans EC2 target

5. Buat tabel — Gabungkan semua metadata yang diidentifikasi pada langkah sebelumnya ke dalam satu tabel.

Nama atribut	Jenis objek	Deskripsi atau tujuan
`wave_ID`	Gelombang sumber	ID gelombang (misalnya: gelombang 10)
`wave_start_date`	Gelombang sumber	Tanggal mulai untuk gelombang
`wave_cutover_date`	Gelombang sumber	Tanggal cutover untuk gelombang
`wave_owner`	Gelombang sumber	Pemilik gelombang
`app_name`	Aplikasi sumber	Nama aplikasi sumber
`app_to_server_mapping`	Aplikasi sumber	Application-to-server hubungan
`app_to_DB_mapping`	Aplikasi sumber	Application-to-database hubungan
`app_to_app_dependencies`	Aplikasi sumber	Dependensi eksternal aplikasi
`AWS_account_ID`	Alat (Layanan Migrasi Aplikasi)	AWS akun untuk meng-host instans EC2 target
`AWS_Region`	Alat (Layanan Migrasi Aplikasi)	AWS Wilayah untuk menjadi tuan rumah instans EC2 target

Nama atribut	Jenis objek	Deskripsi atau tujuan
`replication_server_subnet`	Alat (Layanan Migrasi Aplikasi)	Subnet untuk server replikasi Layanan Migrasi Aplikasi
`replication_server_security_group`	Alat (Layanan Migrasi Aplikasi)	Grup keamanan untuk server replikasi Layanan Migrasi Aplikasi
`server_name`	Server sumber	Nama server sumber
`server_FQDN`	Server sumber	Nama domain yang sepenuhnya memenuhi syarat dari server sumber
`server_OS_family`	Server sumber	Keluarga sistem operasi (OS) dari server sumber (misalnya: Windows atau Linux)
`server_OS_version`	Server sumber	Versi OS dari server sumber (misalnya: Windows Server 2003)
`server_environment`	Server sumber	Lingkungan server sumber (misalnya: pengembangan, produksi, atau pengujian)
`server_tier`	Server sumber	Tingkat server sumber (misalnya: web, database, atau aplikasi)
`CPU`	Server sumber	Jumlah CPUs di server sumber
`RAM`	Server sumber	Ukuran RAM dari server sumber
`disk_size`	Server sumber	Ukuran disk dari server sumber

Nama atribut	Jenis objek	Deskripsi atau tujuan
`target_subnet`	Server target	Subnet dari instans EC2 target
`target_subnet_test`	Server target	Uji subnet dari instans EC2 target
`target_security_group`	Server target	Kelompok keamanan dari instans EC2 target
`target_security_group_test`	Server target	Uji kelompok keamanan dari instans EC2 target
`instance_type`	Server target	Jenis instans dari instans EC2 target
`IAM_role`	Server target	AWS Identity and Access Management (IAM) peran instans EC2 target
`Name`	Server target (tag)	Tag untuk menentukan nama instans EC2 target
`app_owner`	Aplikasi sumber	Pemilik aplikasi sumber
`business_unit`	Server target (tag)	Tag untuk mengidentifikasi unit bisnis untuk instans EC2 target (misalnya: SDM, keuangan, atau TI)
`cost_center`	Server target (tag)	Tag untuk mengidentifikasi pusat biaya untuk instans EC2 target

6. Ulangi — Ulangi proses ini sampai Anda telah mendokumentasikan metadata yang diperlukan untuk setiap pola.

Langkah 2: Membangun penyimpanan metadata dan proses pengumpulan

Pada langkah sebelumnya, Anda menentukan metadata yang diperlukan untuk mendukung migrasi Anda. Pada langkah ini, Anda membangun proses untuk mengumpulkan dan menyimpan metadata. Langkah ini terdiri dari dua tugas:

1. Analisis metadata yang diperlukan dari langkah sebelumnya dan identifikasi sumbernya.
2. Tentukan proses untuk menyimpan dan mengumpulkan metadata secara efisien.

Menganalisis sumber metadata

Ada banyak sumber metadata yang umum. Biasanya, hal pertama yang dapat Anda akses adalah inventaris aset tingkat tinggi, yang biasanya diekspor dari database manajemen konfigurasi (CMDB) atau dari alat lain yang ada. Namun, Anda perlu mengumpulkan metadata dari sumber lain juga, menggunakan proses otomatis dan manual.

Tabel berikut berisi sumber umum, proses pengumpulan standar untuk sumber tersebut, dan jenis metadata umum yang dapat Anda temukan dari sumber tersebut.

Sumber metadata	Jenis koleksi	Jenis metadata
Alat penemuan	Otomatis	Server sumber
CMDB	Otomatis	Server sumber
Inventaris dari alat lain, seperti RVToolsuntuk VMware vSphere	Otomatis	Server sumber
Kuesioner pemilik aplikasi	Manual	Server sumber, server target, gelombang
Wawancara pemilik aplikasi	Manual	Server sumber, server target, gelombang
Dokumentasi desain aplikasi	Manual	Server target
Dokumentasi desain zona pendaratan	Manual	Server target, alat

Setelah mencantumkan semua kemungkinan sumber metadata Anda, Anda menganalisis jenis metadata dan memetakan setiap sumber ke atribut metadata yang Anda identifikasi pada langkah sebelumnya.

1. Dapatkan daftar lengkap atribut metadata dari. Langkah 1: Tentukan metadata yang diperlukan
2. Analisis setiap jenis metadata dan tentukan tipe mana yang tidak dapat diambil menggunakan proses otomatis. Ini biasanya metadata server target dan jenis metadata gelombang karena ini memerlukan keputusan dari pemilik aplikasi. Misalnya, subnet dan grup keamanan mana yang akan Anda gunakan untuk instans EC2 target?
3. Analisis setiap atribut metadata dan petakan ke sumber metadata di tabel sebelumnya. Adalah umum untuk memiliki kombinasi berbagai sumber. Anda dapat menggunakan alat penemuan untuk mengumpulkan beberapa metadata server sumber. Untuk informasi tentang penggunaan alat penemuan untuk mengumpulkan metadata, lihat Memulai penemuan portofolio otomatis di situs web Panduan AWS Preskriptif.
4. Buat tabel untuk memetakan atribut metadata ke jenis dan sumbernya. Tabel berikut adalah contohnya.

Atribut metadata	Jenis metadata	Sumber metadata
`app_name`	Aplikasi sumber	CMDB
`app_owner`	Aplikasi sumber	CMDB
`app_to_server_mapping`	Aplikasi sumber	CMDB, alat penemuan, atau kuesioner pemilik aplikasi
`app_to_DB_mapping`	Aplikasi sumber	CMDB, alat penemuan, atau kuesioner pemilik aplikasi
`app_to_app_dependencies`	Aplikasi sumber	CMDB, alat penemuan, atau kuesioner pemilik aplikasi
`server_name`	Server sumber	CMDB
`server_FQDN`	Server sumber	CMDB
`server_OS_family`	Server sumber	CMDB

Atribut metadata	Jenis metadata	Sumber metadata
`server_IP`	Server sumber	Alat penemuan
`disk_size`	Server sumber	Alat penemuan
`instance_type`	Server target	Alat penemuan
`target_subnet`	Server target	Kuesioner pemilik aplikasi
`target_security_group`	Server target	Kuesioner pemilik aplikasi
`AWS_Region`	Server target	Kuesioner pemilik aplikasi
`AWS_account_ID`	Server target	Kuesioner pemilik aplikasi
`replication_server_subnet`	Alat (Layanan Migrasi Aplikasi)	Dokumentasi desain zona pendaratan
`replication_server_security_group`	Alat (Layanan Migrasi Aplikasi)	Dokumentasi desain zona pendaratan
`Name`	Server target (tag)	Kuesioner pemilik aplikasi
`business_unit`	Server target (tag)	Kuesioner pemilik aplikasi
`cost_center`	Server target (tag)	Kuesioner pemilik aplikasi
`wave_ID`	Perencanaan gelombang	Wawancara pemilik aplikasi
`wave_start_date`	Perencanaan gelombang	Wawancara pemilik aplikasi
`wave_cutover_date`	Perencanaan gelombang	Wawancara pemilik aplikasi

Tentukan satu toko metadata

Setelah memetakan setiap atribut metadata ke sumbernya, Anda menentukan tempat menyimpan metadata. Terlepas dari bagaimana dan di mana Anda menyimpan metadata, Anda hanya perlu

memilih satu repositori. Ini memastikan bahwa Anda memiliki satu sumber kebenaran. Menyimpan metadata di banyak tempat adalah kesalahan umum dalam migrasi besar.

Opsi 1: Simpan metadata dalam spreadsheet di repositori bersama

Meskipun opsi ini mungkin terdengar seperti proses yang sangat manual, ini adalah penyimpanan data paling umum untuk migrasi besar. Hal ini juga umum untuk menyimpan spreadsheet dalam repositori bersama, seperti situs Microsoft. SharePoint

Spreadsheet Microsoft Excel mudah disesuaikan dan tidak membutuhkan waktu lama untuk dibangun. Kerugiannya adalah akan menjadi sangat kompleks jika Anda memiliki banyak metadata dan sulit untuk mengelola hubungan antar aset, seperti antara server, aplikasi, dan database. Tantangan lainnya adalah manajemen versi. Anda perlu membatasi akses tulis hanya untuk beberapa orang, atau Anda perlu menggunakan proses otomatis untuk memperbarui spreadsheet.

Dalam templat buku pedoman portofolio, Anda dapat menggunakan templat Dasbor untuk perencanaan gelombang dan migrasi (format Excel) sebagai titik awal untuk membuat spreadsheet penyimpanan data Anda sendiri.

Opsi 2: Simpan metadata dalam alat yang dibuat khusus

Anda dapat menggunakan alat bawaan, seperti TDS Transition Manager (situs web TDS), untuk menyimpan data Anda, atau Anda dapat membuat alat Anda sendiri. Saat Anda membangun alat Anda sendiri, Anda memerlukan tabel database seperti tab spreadsheet Excel di opsi 1. Contoh:

- Tabel server
- Tabel aplikasi
- Tabel basis data
- Application-to-server dan tabel application-to-database pemetaan
- Tabel perencanaan gelombang
- Tabel kuesioner pemilik aplikasi

Tentukan proses pengumpulan metadata

Pada langkah sebelumnya, Anda memetakan metadata ke sumbernya dan menentukan penyimpanan data tempat Anda akan mengumpulkan metadata. Pada langkah ini, Anda membangun proses untuk mengumpulkan metadata secara efektif. Anda harus meminimalkan copy-and-paste proses manual dan menggunakan otomatisasi untuk mengumpulkan metadata dari setiap sumber. Ada tiga langkah:

1. Buat skrip ekstrak, transformasi, dan muat (ETL) untuk setiap sumber metadata berdasarkan tabel pemetaan metadata.
2. Buat tugas terjadwal yang mengimpor metadata dari setiap sumber secara otomatis secara teratur.
3. Bangun proses ekspor atau berikan akses antarmuka pemrograman aplikasi (API) ke metadata yang disimpan di repositori.

Tabel berikut adalah contoh atribut metadata yang dikumpulkan oleh setiap skrip ETL. Metadata disimpan di lokasi yang Anda tentukan di bagian sebelumnya, seperti spreadsheet atau alat yang dibuat khusus.

Atribut metadata	Jenis metadata	Sumber metadata	Proses pengumpulan
`app_name`	Aplikasi sumber	CMDB	Skrip ETL — CMDB
`app_owner`	Aplikasi sumber	CMDB	Skrip ETL — CMDB
`app_to_server_mapping`	Aplikasi sumber	CMDB	Skrip ETL — CMDB
`app_to_DB_mapping`	Aplikasi sumber	CMDB	Skrip ETL — CMDB
`app_to_app_dependencies`	Aplikasi sumber	Alat penemuan	Skrip ETL — alat penemuan
`server_name`	Server sumber	CMDB	Skrip ETL — CMDB
`server_FQDN`	Server sumber	CMDB	Skrip ETL — CMDB
`server_OS_family`	Server sumber	CMDB	Skrip ETL — CMDB
`server_OS_version`	Server sumber	CMDB	Skrip ETL — CMDB
`disk_size`	Server sumber	Alat penemuan	Skrip ETL — alat penemuan

Atribut metadata	Jenis metadata	Sumber metadata	Proses pengumpulan
`instance_type`	Server target	Alat penemuan	Skrip ETL — alat penemuan
`target_subnet`	Server target	Kuesioner pemilik aplikasi	Skrip ETL - pemilik aplikasi
`target_security_group`	Server target	Kuesioner pemilik aplikasi	Skrip ETL - pemilik aplikasi
`AWS_Region`	Server target	Kuesioner pemilik aplikasi	Skrip ETL - pemilik aplikasi
`AWS_account_ID`	Server target	Kuesioner pemilik aplikasi	Skrip ETL - pemilik aplikasi
`Name`	Server target (tag)	Kuesioner pemilik aplikasi	Skrip ETL - pemilik aplikasi
`business_unit`	Server target (tag)	Kuesioner pemilik aplikasi	Skrip ETL - pemilik aplikasi
`cost_center`	Server target (tag)	Kuesioner pemilik aplikasi	Skrip ETL - pemilik aplikasi
`wave_ID`	Perencanaan gelombang	Kuesioner pemilik aplikasi	Skrip ETL - pemilik aplikasi
`wave_start_date`	Perencanaan gelombang	Kuesioner pemilik aplikasi	Skrip ETL - pemilik aplikasi
`wave_cutover_date`	Perencanaan gelombang	Kuesioner pemilik aplikasi	Skrip ETL - pemilik aplikasi

Langkah 3: Dokumentasikan persyaratan metadata dan proses pengumpulan di runbook

Dalam tugas ini, Anda mendokumentasikan keputusan Anda dalam runbook manajemen metadata. Selama migrasi, alur kerja portofolio Anda mematuhi runbook ini sebagai prosedur standar untuk mengumpulkan dan menyimpan metadata.

1. Dalam template playbook portofolio, buka template Runbook untuk manajemen metadata (format Microsoft Word). Ini berfungsi sebagai titik awal untuk membangun runbook Anda sendiri.
2. Di bagian atribut Metadata, buat tabel atribut metadata untuk setiap pola migrasi, dan isi tabel dengan atribut metadata yang diidentifikasi. Langkah 1: Tentukan metadata yang diperlukan
3. Di bagian Lokasi sumber, dokumentasikan sumber yang Anda identifikasiMenganalisis sumber metadata.
4. Di bagian Petunjuk akses lokasi sumber, dokumentasikan langkah-langkah yang harus diikuti pengguna untuk mengakses lokasi sumber metadata.
5. Di bagian penyimpanan Metadata, dokumentasikan langkah-langkah yang harus diikuti pengguna untuk mengakses penyimpanan metadata yang Anda buat. Tentukan satu toko metadata
6. Di bagian Jenis pengumpulan data, identifikasi proses pengumpulan data yang akan Anda gunakan untuk setiap sumber metadata. Idealnya, Anda harus mengotomatiskan semua koleksi metadata dengan menggunakan skrip otomatisasi.
7. Di bagian Pengumpulan data berdasarkan atribut metadata, untuk setiap atribut metadata, identifikasi hal-hal berikut sesuai dengan instruksi di: Tentukan proses pengumpulan metadata
   a. Jenis metadata
   b. Sumber metadata
   c. Toko metadata
   d. Jenis koleksi
8. Di bagian Kumpulkan metadata, perbarui proses sesuai kebutuhan untuk kasus penggunaan Anda. Ini adalah proses yang diikuti oleh alur kerja portofolio pada tahap implementasi saat mereka mengumpulkan metadata untuk gelombang.
9. Verifikasi bahwa runbook Anda lengkap dan akurat. Runbook ini harus menjadi sumber kebenaran selama migrasi.
10. Bagikan runbook manajemen metadata Anda dengan tim untuk ditinjau.

Kriteria keluar tugas

Lanjutkan ke tugas berikutnya setelah Anda menyelesaikan yang berikut:

- Anda telah menyiapkan satu repositori untuk menyimpan metadata yang dikumpulkan.
- Dalam runbook manajemen metadata Anda, Anda telah mendefinisikan dan mendokumentasikan hal-hal berikut:
  - Atribut metadata yang diperlukan untuk setiap pola migrasi
  - Sumber metadata dan instruksi terperinci tentang cara mengakses setiap sumber
  - Penyimpanan metadata dan instruksi terperinci tentang cara mengaksesnya
  - Proses yang digunakan untuk mengumpulkan metadata
  - Tabel pemetaan yang memetakan atribut metadata ke sumber metadata dan proses pengumpulan

Tugas 3: Mendefinisikan proses prioritas aplikasi

Prioritas aplikasi adalah proses penentuan urutan aplikasi yang harus dimigrasikan ke cloud. Anda menilai prioritas berdasarkan kompleksitas migrasi aplikasi ke cloud dan aturan yang Anda tentukan. Ketika membahas prioritas aplikasi, prioritas tinggi tidak selalu berkorelasi dengan pentingnya aplikasi untuk bisnis. Bahkan, aplikasi yang bisnis kritis biasanya prioritas rendah untuk migrasi karena aplikasi bisnis kritis memiliki risiko yang lebih tinggi. Dalam migrasi besar, Anda memprioritaskan aplikasi dengan kompleksitas rendah yang tidak penting bagi bisnis, dan dengan setiap gelombang, Anda memigrasikan aplikasi yang semakin kompleks atau kritis bisnis.

Dalam migrasi besar, di mana Anda memiliki ratusan aplikasi yang berbaris untuk migrasi, kami tidak menyarankan Anda memprioritaskan dan merencanakan setiap aplikasi sekaligus. Ini adalah salah satu alasan mengapa mendefinisikan proses prioritas aplikasi sangat penting untuk proyek migrasi besar. Untuk mendekati migrasi dengan cara yang gesit, Anda dapat memilih aplikasi prioritas tertinggi (3—10 aplikasi), atau Anda dapat memilih aplikasi yang cukup untuk 3-5 gelombang. Anda kemudian menyelesaikan penemuan aplikasi dan perencanaan gelombang hanya untuk aplikasi yang dipilih. Pendekatan ini menghemat banyak waktu karena prioritas aplikasi dan gelombang sering berubah selama migrasi besar.

Salah satu mitos umum tentang prioritas aplikasi adalah bahwa aplikasi prioritas tertinggi harus berada di gelombang pertama. Ketika Anda melakukan perencanaan gelombang, ada kemungkinan besar bahwa hanya beberapa dari 10 aplikasi prioritas tertinggi yang akan berada di

gelombang pertama karena yang lain tidak siap. Ini bisa karena berbagai alasan yang valid, seperti ketergantungan, kendala bisnis, atau ketersediaan sumber daya. Prioritas aplikasi adalah faktor penting dalam perencanaan gelombang, tetapi seharusnya tidak menjadi satu-satunya faktor yang Anda pertimbangkan.

Dalam tugas ini, Anda menentukan proses dan aturan prioritas aplikasi. Tugas ini terdiri dari langkah-langkah berikut:

- Langkah 1: Tentukan proses prioritas aplikasi
- Langkah 2: Tentukan aturan prioritas aplikasi
- Langkah 3: Selesaikan proses prioritas aplikasi

Bagian selanjutnya membahas penilaian kompleksitas. Buku pedoman ini menyediakan tiga opsi proses untuk memprioritaskan aplikasi, dan dua dari tiga opsi menggunakan penilaian kompleksitas. Untuk informasi selengkapnya tentang opsi proses, lihatLangkah 1: Tentukan proses prioritas aplikasi. Jika Anda berencana menggunakan proses nominasi aplikasi, maka Anda tidak perlu menentukan kriteria penilaian kompleksitas, dan Anda harus melanjutkan langsung ke. Langkah 1: Tentukan proses prioritas aplikasi

Tentang kriteria penilaian kompleksitas

Penilaian kompleksitas adalah proses yang digunakan untuk menilai kesulitan migrasi aplikasi, yang merupakan faktor penting ketika memprioritaskan aplikasi. Penilaian kompleksitas melibatkan evaluasi semua aplikasi terhadap serangkaian kriteria bisnis dan teknis yang sama, yang Anda tentukan. Saat mengevaluasi aplikasi, Anda menetapkan skor untuk setiap kriteria. Ketika Anda menjumlahkan skor kriteria bisnis dan kriteria teknis, Anda mendapatkan skor kompleksitas yang mencerminkan kompleksitas keseluruhan migrasi aplikasi itu. Anda kemudian dapat menggunakan skor kompleksitas saat memprioritaskan aplikasi dan gelombang perencanaan.

Ada dua kategori kriteria penilaian kompleksitas:

- Kriteria bisnis — Kriteria dalam kategori ini berkaitan dengan kompleksitas bisnis dari migrasi aplikasi, seperti risiko jika aplikasi menjadi tidak tersedia, pertimbangan keamanan dan kepatuhan, dan ketersediaan sumber daya.
- Kriteria teknis — Kriteria dalam kategori ini berkaitan dengan kompleksitas teknis migrasi aplikasi, seperti sistem operasi dan versinya, jumlah server dan pengguna, dan strategi migrasi.

Anda harus menentukan kriteria penilaian yang sesuai untuk kasus penggunaan Anda. Jika Anda menilai kompleksitas aplikasi secara manual, dalam templat buku pedoman portofolio, templat lembar Skor untuk kompleksitas aplikasi (format Microsoft Excel) berisi serangkaian kriteria dan nilai skor standar. Anda mungkin ingin memulai dengan nilai-nilai ini dan kemudian menyesuaikannya untuk kasus penggunaan Anda. Jika Anda menggunakan alat penemuan untuk prioritas aplikasi, alat ini biasanya menyertakan serangkaian kriteria standar, dan Anda dapat menambahkan, menghapus, atau memodifikasi kriteria, dan Anda dapat menimbangnya sesuai dengan kebutuhan Anda.
Saat Anda menetapkan kriteria, gunakan pertanyaan di dua bagian berikutnya untuk membantu menyempurnakan kriteria Anda.

Kriteria bisnis

Berikut ini adalah kriteria bisnis yang biasa digunakan dalam penilaian kompleksitas.

Kriteria bisnis	Deskripsi
Dampak bisnis	Menilai dampaknya terhadap bisnis jika aplikasi ini tidak tersedia: • Apakah itu memiliki dampak finansial? • Apakah itu memiliki dampak operasi? • Apakah itu berdampak pada pengalaman pelanggan? • Apakah itu berdampak pada produk atau acara perusahaan?
Ketersediaan staf	Selama migrasi, Anda mungkin memerluka n bantuan dari pemilik aplikasi, pakar materi pelajaran (UKM), administrator jaringan atau infrastruktur, penguji, dan pengembang. Menilai ketersediaan sumber daya ini untuk membantu Anda selama migrasi: • Apakah staf ini akan tersedia selama migrasi untuk membantu dan memberikan panduan kepada tim migrasi?

Kriteria bisnis	Deskripsi
	• Apakah staf ini akan tersedia untuk menguji dan memvalidasi aplikasi setelah dimigrasi kan? • Apakah staf ini akan tersedia untuk menyediakan alamat IP atau port yang diperlukan untuk menjalankan aplikasi di lingkungan target?
Kompleksitas bisnis	Memiliki banyak pemangku kepentingan yang saling bergantung dan saling berhubung an, sistem teknologi informasi, dan struktur organisasi dapat meningkatkan kompleksitas bisnis. Menilai kompleksitas bisnis sebagai berikut: • Berapa lama waktu yang dibutuhkan bisnis untuk menyetujui infrastruktur dan perubahan jaringan, seperti perubahan firewall atau penyediaan instance baru? • Berapa lama waktu yang dibutuhkan bisnis untuk menyetujui instalasi perangkat lunak atau alat baru di server mereka, seperti alat penemuan?

Kriteria bisnis	Deskripsi
Kesiapan	Menilai apakah aplikasi siap untuk migrasi sebagai berikut: • Apakah aplikasi saat ini sedang menjalani atau dijadwalkan menjalani penyegaran teknologi? • Apakah pemeliharaan dijadwalkan, dan apakah akan tumpang tindih dengan migrasi terjadwal? • Apakah aplikasi dijadwalkan untuk dinonakti fkan? • Apakah aplikasi saat ini sedang ditingkat kan, dan apakah ada fitur baru yang sedang dikembangkan atau diintegrasikan?

Kriteria bisnis	Deskripsi
Keamanan	Menilai kompleksitas persyaratan keamanan aplikasi dan kebijakan keamanan sebagai berikut: • Apakah Anda perlu menyediakan alamat IP dan port untuk mengakses aplikasi? • Apakah aplikasi memerlukan perlindungan infrastruktur? • Apakah aplikasi memerlukan perlindungan data? • Apakah manajemen kunci diperlukan? • Apakah aplikasi memerlukan kebijakan manajemen akses khusus? • Apakah aplikasi memerlukan pemantauan atau pencatatan? • Apakah aplikasi memerlukan proses respons insiden dan otomatisasi? • Apakah peringatan dan pemberitahuan diperlukan untuk aplikasi ini?

Kriteria bisnis	Deskripsi
Kepatuhan	Persyaratan kepatuhan mungkin berlaku untuk aplikasi, seperti undang-undang, peraturan, dan pedoman yang disediakan oleh negara bagian, industri bisnis, atau kebijakan perusahaan. Menilai kompleksitas persyaratan kepatuhan aplikasi sebagai berikut: • Apakah ada persyaratan privasi dan residensi data? • Haruskah data yang diam dalam aplikasi dienkripsi? • Haruskah data dalam perjalanan ke atau dari aplikasi dienkripsi? • Apakah pencatatan audit diperlukan? • Apakah aplikasi harus sesuai dengan standar akuntansi dan keuangan, seperti System and Organization Controls (SOC)? • Apakah aplikasi harus sesuai dengan standar keamanan pembayaran, seperti Payment Card Industry (PCI)? • Apakah aplikasi harus mematuhi peraturan informasi kesehatan pasien, seperti Health Insurance Portability and Accountability Act (HIPAA)? • Apakah aplikasi harus sesuai dengan program keamanan cloud publik, seperti Program Manajemen dan Penilaian Keamanan Sistem Informasi (ISMAP)?

Kriteria bisnis	Deskripsi
Pengetahuan aplikasi	Apakah seseorang dalam organisasi, seperti pemilik aplikasi, memiliki pengetahuan, keterampilan, dan pengalaman untuk memelihara, mengintegrasikan, memecahkan masalah, dan memperbaiki masalah? Dan apakah Anda dapat memperluas aplikasi untuk memenuhi permintaan bisnis?
Keterampilan migrasi	Apakah staf di organisasi Anda memiliki keterampilan untuk memigrasikan beban kerja ke lingkungan target?

Kriteria teknis

Berikut ini adalah kriteria teknis yang biasa digunakan dalam penilaian kompleksitas.

Kriteria teknis	Deskripsi
Penyimpanan	Nilai penyimpanan aplikasi saat ini sebagai berikut: • Di mana aplikasi saat ini disimpan? Contohnya termasuk penyimpanan terlampir jaringan (NAS), jaringan area penyimpanan (SAN), atau drive lokal. • Berapa ukuran total penyimpanan saat ini?
Jumlah pengguna	Berapa banyak pengguna yang dimiliki aplikasi ini? Anda dapat menggunakan log atau perkiraan yang sebenarnya.
Jumlah server	Berapa banyak server yang ada di tumpukan aplikasi?

Kriteria teknis	Deskripsi
Konektivitas	Menilai bagaimana aplikasi ini terhubung dengan orang lain di organisasi Anda sebagai berikut: • Berapa banyak aplikasi lain yang bergantung pada aplikasi ini? • Apa dampaknya terhadap aplikasi lain jika aplikasi ini tidak tersedia?
Aplikasi OS dan versi	Menilai sistem operasi (OS) dan versi server aplikasi sebagai berikut: • Apakah versi OS server tidak lagi didukung? • Apakah server menjalankan OS Unix atau Linux? • Apakah server menjalankan OS Windows Server? • Apakah aplikasi di mainframe atau di server midrange?
Dependensi aplikasi	Menilai bagaimana aplikasi ini bergantung pada sumber daya lain di lingkungan Anda: • Sumber daya apa yang bergantung pada aplikasi ini? Sumber daya dapat berupa aplikasi lain, komponen, layanan khusus OS (seperti pendaftar atau server web), atau perpustakaan. • Apa dampaknya terhadap aplikasi ini jika satu atau lebih sumber daya tersebut tidak tersedia?

Kriteria teknis	Deskripsi
Migrasi data	Menilai apakah Anda perlu memigrasi data atau file untuk aplikasi ini: • Seberapa kompleks migrasi data? • Seberapa kompleks migrasi file?
Strategi migrasi	Menilai kompleksitas strategi migrasi yang dipilih. Untuk informasi selengkapnya tentang strategi migrasi, lihat Panduan untuk migrasi AWS besar.
COTS atau kustom	Menilai apakah aplikasi dibuat khusus atau komersial off-the-shelf (COTS) sebagai berikut: • Apakah Anda memiliki versi terbaru dari kode sumber? • Apakah vendor aplikasi didukung? • Apakah aplikasi tersebut dialihdayakan?

Langkah 1: Tentukan proses prioritas aplikasi

Buku pedoman ini mencakup tiga opsi proses untuk memprioritaskan aplikasi. Anda dapat memilih salah satu opsi, atau Anda mungkin memutuskan untuk menggabungkan dua atau lebih dan membangun proses kustom. Evaluasi kasus penggunaan Anda dan tentukan mana dari berikut ini yang paling cocok untuk lingkungan Anda:

- Opsi 1: Penilaian kompleksitas manual— Ini adalah proses prioritas manual yang dapat diselesaikan oleh individu atau dalam sesi bergaya lokakarya. Dalam proses ini, Anda menggunakan kriteria penilaian kompleksitas untuk menilai kesulitan migrasi setiap aplikasi, yang merupakan faktor penting dalam memprioritaskan aplikasi. Proses manual ini sangat cocok untuk migrasi besar karena memberikan pendekatan kuantitatif yang konsisten untuk memprioritaskan portofolio aplikasi yang besar. Namun, mengevaluasi setiap aplikasi terhadap serangkaian kriteria yang ditentukan dapat menjadi proses yang lebih lambat jika dibandingkan dengan dua opsi lainnya.

- Opsi 2: Nominasi aplikasi— Ini adalah proses prioritas manual yang biasanya menyelesaikan sesi bergaya lokakarya. Dalam proses ini, pemilik aplikasi menominasikan aplikasi untuk migrasi. Agar berhasil, proses ini mengharuskan pemilik aplikasi memiliki pengetahuan yang komprehensif tentang aplikasi masing-masing. Proses ini disarankan jika waktu merupakan faktor dan Anda perlu memprioritaskan aplikasi dengan cepat.
- Opsi 3: Alat penemuan— Ini adalah proses prioritas otomatis. Jika alat penemuan di lingkungan Anda memiliki fitur otomatis untuk penilaian atau prioritas kompleksitas aplikasi otomatis, menggunakan fitur ini dapat menghemat waktu dan mempercepat proses prioritas aplikasi. Dalam proses ini, Anda biasanya menentukan kriteria dalam parameter alat penemuan, dan kemudian alat menganalisis aplikasi dan memberikan skor kompleksitas akhir. Sebelum memilih opsi ini, jelajahi fitur yang tersedia di alat penemuan Anda dan verifikasi bahwa Anda dapat menyesuaikannya untuk memenuhi kebutuhan kasus penggunaan Anda.

Opsi 1: Penilaian kompleksitas manual

Dalam proses prioritas aplikasi manual ini, Anda menggunakan lembar kerja untuk mengevaluasi aplikasi terhadap serangkaian kriteria penilaian kompleksitas yang ditentukan. Kami menyarankan Anda menyelesaikan lembar kerja dalam sesi bergaya lokakarya, atau seorang individu dapat menyelesaikan lembar kerja dengan berkolaborasi dengan pemangku kepentingan. Anda kemudian menggunakan skor kompleksitas akhir dan aturan prioritas aplikasi untuk menentukan prioritas aplikasi. Dari proses manual, ini memberikan pendekatan kuantitatif yang paling konsisten untuk menentukan kompleksitas aplikasi dan menggunakan informasi tersebut untuk memprioritaskan aplikasi.

Untuk langkah-langkah penilaian kompleksitas dalam proses ini, kami sarankan Anda menggunakan template lembar Skor untuk kompleksitas aplikasi (format Excel), tersedia dalam templat buku pedoman portofolio. Template ini mencakup kriteria bisnis dan teknis yang telah ditentukan. Anda dapat menambahkan, menghapus, atau memodifikasi kriteria ini, atau Anda dapat menyesuaikan nilai penilaian. Misalnya, Anda mungkin lebih suka rentang skor 1-10 daripada 1-5. Perhatikan hal berikut tentang template yang disediakan:

- Anda dapat mengarahkan kursor ke setiap kriteria untuk deskripsi itu.
- Ketika Anda terbiasa dengan template, Anda harus menghapus contoh. Ini hanya untuk tujuan demonstrasi.

Perbarui lembar skor kompleksitas selama tahap inisialisasi dan implementasi migrasi. Anda dapat mengubah skor saat Anda maju melalui penilaian portofolio. Aplikasi deep dive adalah waktu yang

umum untuk memperbarui lembar skor karena Anda mempelajari lebih lanjut tentang setiap aplikasi saat tim memeriksanya secara rinci. Selama migrasi, Anda juga dapat mengubah prioritas aplikasi jika mengalami masalah, seperti dependensi yang belum ditemukan dan batasan yang mencegah Anda memigrasi aplikasi. Dengan mempertahankan lembar skor selama migrasi, Anda dapat memprioritaskan aplikasi dengan akurasi yang lebih besar.

Dokumentasikan proses prioritas aplikasi Anda sebagai berikut:

1. Dalam templat buku pedoman portofolio, buka template lembar Skor untuk kompleksitas aplikasi.
2. Pada lembar Aplikasi, tambahkan, ubah, atau hapus kriteria yang sesuai untuk migrasi Anda. Saat memodifikasi kriteria, lakukan hal berikut:
   - Tinjau panduan di Tentang kriteria penilaian kompleksitas bagian buku pedoman ini.
   - Pertimbangkan dampak setiap kriteria terhadap durasi, sumber daya, dan biaya migrasi.
   - Untuk skor kompleksitas yang andal, sertakan kriteria yang mewakili berbagai tingkat kompleksitas migrasi di organisasi Anda.
3. Pada lembar panduan Penilaian, perbarui nilai dan kriteria default sesuai kebutuhan untuk kasus penggunaan Anda.
4. Simpan lembar skor.
5. Buka runbook prioritas aplikasi Anda.
6. Di bagian Kriteria penilaian kompleksitas aplikasi, perbarui bagian untuk mencerminkan lokasi lembar skor Anda.
7. Di bagian Proses prioritas Aplikasi, lakukan hal berikut:
   a. Keep Option 1: Penilaian kompleksitas manual dan hapus opsi lainnya.
   b. Ubah proses sesuai kebutuhan untuk kasus penggunaan Anda.
   c. Hapus judul apa pun di bagian ini yang berisi kata Opsi. Membiarkan ini di runbook mungkin membingungkan pengguna untuk berpikir bahwa prosesnya opsional atau beberapa opsi tersedia.
   d. Simpan runbook prioritas aplikasi Anda.

Opsi 2: Nominasi aplikasi

Proses prioritas aplikasi manual ini adalah pendekatan termudah dan tercepat untuk memprioritaskan aplikasi. Dalam proses ini, Anda meminta pemilik aplikasi untuk menominasikan aplikasi yang dapat dengan mudah dimigrasikan ke cloud. Anda dan pemilik aplikasi kemudian dapat dengan cepat

memprioritaskan aplikasi karena Anda sudah memiliki pengetahuan mendalam tentang aplikasi yang dinominasikan. Kami menyarankan Anda bekerja dengan pemangku kepentingan dalam sesi bergaya lokakarya, tetapi Anda juga dapat berkolaborasi melalui email, dokumentasi bersama, dan platform komunikasi lainnya.

Selama proses nominasi, Anda memasukkan aplikasi yang dinominasikan ke dalam template lembar Skor untuk kompleksitas aplikasi (format Excel), termasuk dalam templat buku pedoman portofolio. Anda tidak akan menggunakan semua fitur penilaian dan kriteria dalam template ini, tetapi kami sarankan menggunakan lembar ini untuk mencatat keputusan nominasi dan prioritas.

Dalam beberapa situasi, proses nominasi aplikasi digunakan untuk mempercepat prioritas, dan lembar skor mungkin tidak diperlukan. Misalnya, jika Anda memprioritaskan hanya segelintir aplikasi atau jika pemilik aplikasi sangat berpengetahuan tentang aplikasi mereka, pemilik aplikasi dan pemangku kepentingan dapat memprioritaskan aplikasi berdasarkan pengetahuan dan pengalaman mereka. Dalam hal ini, mereka dapat melewati menggunakan lembar skor, dan melanjutkan langsung ke prioritas.

Dokumentasikan proses prioritas aplikasi Anda sebagai berikut:

1. Buka runbook prioritas aplikasi Anda.
2. Hapus bagian Kriteria penilaian kompleksitas aplikasi. Proses ini tidak menggunakan penilaian kompleksitas aplikasi.
3. Di bagian Proses prioritas Aplikasi, lakukan hal berikut:
   a. Keep Option 2: Nominasi aplikasi dan hapus opsi lainnya.
   b. Ubah proses sesuai kebutuhan untuk kasus penggunaan Anda.
   c. Hapus judul apa pun di bagian ini yang berisi kata Opsi. Membiarkan ini di runbook mungkin membingungkan pengguna untuk berpikir bahwa prosesnya opsional atau beberapa opsi tersedia.
4. Simpan runbook prioritas aplikasi Anda.

Opsi 3: Alat penemuan

Jika alat penemuan Anda memiliki fitur untuk penilaian kompleksitas atau prioritas aplikasi, proses otomatis ini memerlukan sedikit sumber daya dan dapat mempercepat proses prioritas aplikasi. Anda menyesuaikan kriteria dalam alat penemuan untuk kasus penggunaan Anda, dan kemudian alat penemuan secara otomatis menganalisis aplikasi dan memberikan skor kompleksitas akhir. Karena alat ini sudah memiliki semua metadata aplikasi, Anda tidak perlu memasukkannya.

Misalnya, alat penemuan Flexera One Cloud Migration and Modernization (sebelumnya Flexera Foundation dan CloudScape) memiliki fitur penilaian kompleksitas yang disebut Optimization Scorecard. Fitur ini memungkinkan Anda untuk memilih kriteria yang ingin Anda sertakan dalam penilaian dan menimbang setiap kriteria berdasarkan preferensi Anda. Setelah penemuan data selesai, alat penemuan menganalisis data berdasarkan kriteria tertimbang yang Anda berikan dan, menggunakan rumus kepemilikan alat, menghasilkan skor kompleksitas akhir. Untuk informasi selengkapnya, lihat Foundation and CloudScape User Guide (dokumentasi Flexera) dan Optimization Scorecard (dokumentasi Flexera).

Kerugian dari proses ini adalah membutuhkan waktu (4-8 minggu) untuk menyiapkan alat pemindaian untuk alat penemuan tanpa agen di lingkungan Anda atau untuk menginstal agen ke semua beban kerja dalam ruang lingkup. Sebelum Anda dapat menggunakan fitur penilaian di alat penemuan Anda, Anda harus memberikan waktu tambahan (4—12 minggu) agar alat penemuan mengumpulkan metadata dengan memindai beban kerja aplikasi dan melakukan analisis tumpukan aplikasi. Namun, Anda mungkin menemukan bahwa waktu tambahan yang diperlukan untuk mengonfigurasi alat penemuan mungkin dipulihkan dengan mengurangi jumlah waktu dan sumber daya yang dibutuhkan untuk pengumpulan metadata dan prioritas aplikasi. Misalnya, jika data alat penemuan masih terkini, alur kerja portofolio dapat menggunakan kembali alat penemuan dan datanya dari fase mobilisasi untuk mengidentifikasi aplikasi percontohan.

> **Note**
>
> Jika Anda menggunakan proses alat penemuan, Anda masih dapat menggunakan templat lembar Skor manual untuk kompleksitas aplikasi untuk menganalisis aplikasi terhadap serangkaian kriteria yang berbeda. Informasi tambahan ini dapat membantu Anda menyempurnakan prioritas aplikasi Anda.

Dokumentasikan proses prioritas aplikasi Anda sebagai berikut:

1. Jika Anda belum melakukannya, siapkan alat penemuan di lingkungan Anda. Untuk informasi selengkapnya, lihat Memulai penemuan portofolio otomatis di situs web Panduan AWS Preskriptif.
2. Sesuaikan penilaian kompleksitas atau kriteria prioritas aplikasi di alat penemuan Anda sesuai dengan instruksi untuk alat Anda. Untuk informasi selengkapnya tentang memilih kriteria, lihatTentang kriteria penilaian kompleksitas.
3. Buka runbook prioritas aplikasi Anda.

4. Di bagian Kriteria penilaian kompleksitas aplikasi, perbarui bagian untuk mencerminkan bahwa kriteria penilaian ditentukan dalam alat penemuan. <your discovery tool>Contoh: Kriteria penilaian kompleksitas didefinisikan dalam.
5. Di bagian Proses prioritas Aplikasi, lakukan hal berikut:
   a. Keep Option 3: Alat penemuan dan hapus opsi lainnya.
   b. Ubah proses sesuai kebutuhan untuk kasus penggunaan Anda. Penting bagi Anda untuk menyertakan step-by-step instruksi tentang cara membuat laporan dengan skor kompleksitas. Jika tersedia, Anda dapat menyertakan tautan ke panduan pengguna.
   c. Hapus judul apa pun di bagian ini yang berisi kata Opsi. Membiarkan ini di runbook mungkin membingungkan pengguna untuk berpikir bahwa prosesnya opsional atau beberapa opsi tersedia.
6. Simpan runbook prioritas aplikasi Anda.

Langkah 2: Tentukan aturan prioritas aplikasi

Pada langkah ini, Anda menentukan aturan prioritas aplikasi, yang membantu Anda menentukan urutan migrasi aplikasi. Meskipun skor kompleksitas aplikasi merupakan faktor penting dalam memprioritaskan aplikasi dan gelombang perencanaan, faktor bisnis dan teknologi juga harus dipertimbangkan. Anda membuat aturan untuk menilai prioritas setiap aplikasi dan membantu Anda menjadwalkan aplikasi dalam gelombang yang sesuai. Aturan bisnis dan teknologi yang umum meliputi:

- Menentukan urutan dan jadwal untuk migrasi pusat data
- Memprioritaskan unit bisnis
- Menangkap tenggat waktu untuk aplikasi bisnis penting

Tentukan aturan prioritas aplikasi Anda sebagai berikut:

1. Buka runbook prioritas aplikasi Anda.
2. Di bagian Aturan prioritas aplikasi, tambahkan aturan khusus untuk migrasi Anda.
3. Simpan runbook prioritas aplikasi.
4. Pertahankan aturan dalam runbook prioritas aplikasi. Aturan dapat berubah sewaktu-waktu saat kemajuan migrasi, perubahan ruang lingkup, atau jadwal bergeser.

Berikut ini adalah contoh seperangkat aturan prioritas aplikasi.

Prioritas	Aturan
1	Aplikasi di pusat data New York harus selalu memiliki prioritas lebih tinggi daripada aplikasi di pusat data Texas.
2	Aplikasi di departemen TI harus selalu memiliki prioritas lebih tinggi daripada aplikasi di departemen Pemasaran.
3	Aplikasi dengan skor kompleksitas tinggi harus memiliki prioritas yang lebih tinggi.
4	Aplikasi SAP perlu dimigrasikan sebelum akhir tahun.

Langkah 3: Selesaikan proses prioritas aplikasi

Sekarang, Anda menentukan bagaimana alur kerja portofolio menggunakan aturan dan proses untuk memprioritaskan aplikasi. Ini adalah proses yang direferensikan oleh alur kerja portofolio dalam tahap implementasi migrasi.

Sesuaikan proses ini di runbook prioritas aplikasi sebagai berikut:

1. Buka runbook prioritas aplikasi Anda.
2. Di bagian Tahap 2: Prioritaskan aplikasi, modifikasi proses yang sesuai untuk kasus penggunaan dan lingkungan Anda.
3. Simpan runbook prioritas aplikasi.

Kriteria keluar tugas

Lanjutkan ke tugas berikutnya ketika Anda telah menyelesaikan yang berikut:

- Anda telah memilih proses prioritas aplikasi dari opsi yang tersedia.
- Anda telah mendokumentasikan hal-hal berikut dalam runbook prioritas aplikasi Anda:

  - Kriteria penilaian kompleksitas aplikasi (jika ada)
  - Proses prioritas aplikasi
  - Aturan prioritas aplikasi
- Anda telah memperbarui bagian Tahap 2: Prioritaskan aplikasi dari runbook aplikasi Anda.

Tugas 4: Mendefinisikan proses penyelaman mendalam aplikasi

Sekarang setelah Anda selesai menetapkan aturan dan proses untuk prioritas aplikasi, Anda melakukan penyelaman mendalam aplikasi yang akan membantu Anda menyempurnakan prioritas setiap aplikasi. Anda melakukan penyelaman mendalam aplikasi pada satu aplikasi pada satu waktu, dalam urutan dari prioritas tertinggi hingga terendah. Untuk proyek dengan beberapa tim portofolio, setiap tim dapat melakukan penyelaman mendalam aplikasi pada aplikasi yang berbeda secara bersamaan.

Selama deep dive, Anda mungkin mengalami beberapa masalah tak terduga, seperti dependensi, yang memengaruhi kompleksitas migrasi aplikasi. Ketika ini terjadi, Anda harus memodifikasi kriteria penilaian kompleksitas yang Anda tentukan pada langkah sebelumnya dan memperbarui lembar skor yang sesuai untuk mendapatkan skor kompleksitas yang lebih akurat untuk aplikasi masa depan. Anda kemudian dapat memperbarui prioritas aplikasi dengan menggunakan skor kompleksitas baru.

Tugas ini terdiri dari langkah-langkah berikut:

- Langkah 1: Tentukan proses lokakarya aplikasi
- Langkah 2: Tentukan proses pemetaan aplikasi
- Langkah 3: (Opsional) Tentukan status target aplikasi
- Langkah 4: Selesaikan proses penyelaman mendalam aplikasi

Langkah 1: Tentukan proses lokakarya aplikasi

Proses lokakarya adalah salah satu pendekatan paling efisien untuk aplikasi deep dive. Dengan menggunakan proses ini, Anda berkolaborasi dengan pemangku kepentingan, pemilik aplikasi, dan tim portofolio untuk menilai dan menganalisis aplikasi. Tujuannya adalah untuk memahami dengan jelas keadaan aplikasi saat ini, termasuk arsitektur, tujuan bisnis, dependensi, dan lingkungannya. Anda kemudian menggunakan informasi terperinci ini tentang ukuran dan kompleksitas aplikasi untuk merancang status target aplikasi.

Setiap lokakarya biasanya berlangsung 1—8 jam, meskipun Anda mungkin menemukan bahwa waktu tambahan diperlukan untuk aplikasi dengan kompleksitas tinggi. Anda juga dapat memecah lokakarya menjadi beberapa pertemuan, tergantung pada ketersediaan sumber daya, preferensi Anda, dan ukuran dan kompleksitas aplikasi.

Identifikasi hasil yang diharapkan

Sebelum melakukan lokakarya aplikasi, Anda harus menetapkan agenda dan menentukan hasil yang diharapkan dari lokakarya, atau informasi yang perlu Anda kumpulkan di lokakarya. Hal ini memungkinkan peserta lokakarya untuk mempersiapkan lokakarya, membantu menjaga pertemuan tetap tepat sasaran, dan memastikan bahwa pada akhir lokakarya, Anda memiliki semua informasi yang diperlukan untuk memprioritaskan, merencanakan gelombang, dan memigrasikan aplikasi.

Kami menyarankan Anda menentukan serangkaian standar hasil yang diharapkan dan mendokumentasikannya dalam runbook prioritas aplikasi Anda. Saat menyiapkan lokakarya, Anda menggunakan hasil standar yang diharapkan dan menambahkan yang baru untuk aplikasi tertentu.

Tentukan set standar hasil yang diharapkan untuk lokakarya aplikasi sebagai berikut:

1. Buka runbook prioritas aplikasi Anda.
2. Di bagian hasil lokakarya Aplikasi yang diharapkan, tetapkan serangkaian standar hasil yang diharapkan untuk lokakarya aplikasi. Saat menyiapkan lokakarya, Anda dapat menyesuaikannya untuk kebutuhan spesifik aplikasi.
3. Simpan runbook prioritas aplikasi.
4. Pertahankan hasil yang diharapkan dalam runbook prioritas aplikasi. Saat Anda melakukan lokakarya aplikasi dan melanjutkan penilaian portofolio dan perencanaan gelombang, Anda dapat mengidentifikasi hasil baru yang diharapkan atau menyempurnakan hasil yang ada.

Berikut ini adalah contoh hasil yang diharapkan untuk lokakarya aplikasi.

Prioritas	Hasil yang diharapkan dari lokakarya aplikasi
1	Pemahaman yang jelas tentang arsitektu r aplikasi saat ini, termasuk server terkait, dependensi, lingkungan, dan tingkat aplikasi.

Prioritas	Hasil yang diharapkan dari lokakarya aplikasi
2	Tim telah mengumpulkan metadata untuk mendukung desain arsitektur target. Metadata berikut diperlukan: • ID AWS akun target • AWS Wilayah Target • Subnet target • Menargetkan grup keamanan
3	Kuesioner pemilik aplikasi lengkap, dan semua pertanyaan kunci dijawab.
4	Tim telah mengumpulkan semua dokumenta si aplikasi, seperti panduan pengguna, dokumentasi arsitektur aplikasi, dokumenta si pengujian, dokumentasi desain, dan dokumentasi antarmuka pemrograman aplikasi (API).

Tentukan aturan lokakarya aplikasi

Sebelum melakukan lokakarya aplikasi, Anda harus menentukan aturan yang akan mengatur lokakarya Anda. Aturan umum termasuk panjang lokakarya, alat yang mungkin diperlukan di bengkel, dan pertimbangan penjadwalan atau tenggat waktu yang perlu dipertimbangkan. Ini membantu menjaga rapat tetap tepat sasaran dan memastikan bahwa keputusan yang dibuat di lokakarya selaras dengan jadwal migrasi.

Kami menyarankan Anda mendokumentasikan aturan lokakarya aplikasi di runbook prioritas aplikasi Anda.

Tentukan aturan lokakarya aplikasi Anda sebagai berikut:

1. Buka runbook prioritas aplikasi Anda.
2. Di bagian Aturan lokakarya aplikasi, tentukan aturan yang mengatur lokakarya Anda.
3. Simpan runbook prioritas aplikasi.

4. Pertahankan aturan dalam runbook prioritas aplikasi. Saat Anda melakukan lokakarya aplikasi dan melanjutkan penilaian portofolio dan perencanaan gelombang, Anda dapat mengidentifikasi aturan baru atau menyempurnakan aturan yang sudah ada.

Berikut ini adalah contoh aturan untuk lokakarya aplikasi.

Prioritas	Aturan lokakarya
1	Lokakarya harus dijadwalkan maksimal 2 jam per sesi pada hari Selasa dan Kamis.
2	Ada pembekuan infrastruktur yang dijadwalkan 1 Desember—15 Januari.
3	Ada lokakarya langsung untuk alat migrasi.
4	Kontrak pusat data akan berakhir pada 31 Maret. Beban kerja harus dievakuasi pada 31 Maret untuk menghindari hukuman dan perpanjangan kontrak yang mahal.
5	Aplikasi biometrik dan aplikasi waktu dan kehadiran akan dipertahankan.

Tentukan proses lokakarya aplikasi

Penting bagi Anda untuk menentukan proses standar untuk melakukan lokakarya aplikasi. Ini memastikan pengalaman yang konsisten dan menetapkan harapan bagi peserta lokakarya, yang dapat meningkatkan efisiensi lokakarya.

Ada tiga tahap proses lokakarya aplikasi:

- Mempersiapkan lokakarya — Mempersiapkan lokakarya membantu memastikan bahwa sesi berjalan lancar dan peserta tahu apa yang diharapkan. Untuk mempersiapkan lokakarya, Anda membuat agenda dan menentukan hasil yang diharapkan, Anda mengidentifikasi peserta, alat, dan informasi yang dibutuhkan dalam lokakarya, dan Anda menjadwalkan lokakarya. Menjadwalkan lokakarya setidaknya satu minggu sebelumnya memberi tim waktu untuk memblokir kalender mereka, mempersiapkan lokakarya, dan mengumpulkan informasi yang berguna.

- Melakukan lokakarya — Saat melakukan lokakarya, Anda membatasi diskusi pada item dalam agenda dan memastikan bahwa Anda memenuhi hasil yang diharapkan. Perhatikan topik yang menurut Anda bermanfaat tetapi tidak termasuk dalam agenda Anda. Akan sangat membantu untuk merekam lokakarya.
- Selesaikan hasil lokakarya — Setelah lokakarya, tim Anda harus memiliki pemahaman yang jelas tentang keadaan aplikasi saat ini dan potensi titik nyeri, risiko, tantangan, dan penghambat yang dapat memengaruhi prioritas dan migrasi. Tugas umum setelah lokakarya meliputi: memprioritaskan ulang aplikasi, menyusun status aplikasi di masa depan, dan memperbarui runbook dengan hasil, aturan, atau perubahan proses yang diharapkan yang mungkin membantu dalam lokakarya berikutnya.

Template Runbook untuk prioritas aplikasi mencakup step-by-step proses standar untuk mempersiapkan, melakukan, dan menyelesaikan lokakarya aplikasi. Tentukan proses lokakarya aplikasi Anda sebagai berikut:

1. Buka runbook prioritas aplikasi Anda.
2. Di bagian Proses lokakarya Aplikasi, modifikasi proses standar untuk memenuhi kebutuhan kasus penggunaan Anda.
3. Simpan runbook prioritas aplikasi.
4. Pertahankan proses dalam runbook prioritas aplikasi. Saat Anda melakukan lokakarya aplikasi, Anda dapat mengidentifikasi perubahan yang ingin Anda lakukan pada proses ini.

Kriteria keluar langkah

- Anda telah menentukan agenda lokakarya dan sumber daya serta artefak yang diperlukan untuk mendukung lokakarya.
- Anda telah menentukan hasil yang diharapkan dari lokakarya dan mengidentifikasi informasi yang perlu Anda kumpulkan di bengkel.
- Anda telah melakukan uji coba proses lokakarya dan memiliki informasi yang diperlukan untuk mendukung pemetaan aplikasi dan merancang status target.
- Anda telah mendokumentasikan hal-hal berikut dalam runbook prioritas aplikasi Anda:
  - Lokakarya aplikasi hasil yang diharapkan
  - Aturan lokakarya aplikasi
  - Proses lokakarya aplikasi

Langkah 2: Tentukan proses pemetaan aplikasi

Pemetaan aplikasi adalah proses menetapkan setiap aplikasi ke pola migrasi, yang Anda identifikasi dan validasi. Langkah 4: Validasi pola migrasi Pada langkah ini, Anda menentukan aturan yang akan Anda gunakan untuk mengevaluasi aplikasi. Anda kemudian menentukan proses yang akan Anda gunakan untuk mengevaluasi setiap aplikasi. Memetakan setiap aplikasi ke kasus penggunaan pola migrasi membantu Anda mengidentifikasi alat migrasi, keterampilan apa pun yang diperlukan untuk menyelesaikan migrasi, dan kompleksitas pola migrasi.

Anda tidak selalu memigrasikan aplikasi ke satu pola. Untuk informasi selengkapnya tentang kapan Anda mungkin memerlukan beberapa pola untuk aplikasi yang sama, lihat Tentukan proses pemetaan aplikasi nanti di bagian ini.

Aturan pemetaan aplikasi

Aturan pemetaan aplikasi membantu Anda mengevaluasi aplikasi dan mengidentifikasi pola migrasi yang sesuai. Setiap aturan terdiri dari informasi apa pun yang harus Anda gunakan untuk mengevaluasi aplikasi dan mencocokkan kriteria untuk pola tersebut.

Dalam template playbook portofolio, template Runbook untuk prioritas aplikasi menyertakan bagian untuk mendokumentasikan aturan pemetaan aplikasi Anda. Tentukan proses Anda sebagai berikut:

1. Buka runbook prioritas aplikasi Anda.
2. Di bagian Aturan pemetaan aplikasi, tentukan aturan pemetaan aplikasi Anda.
3. Simpan runbook prioritas aplikasi.
4. Pertahankan aturan dalam runbook prioritas aplikasi.

Tabel berikut memberikan contoh aturan pemetaan aplikasi.

> **Note**
>
> Pola IDs dan nama dalam tabel ini sesuai dengan pola sampel diLangkah 4: Validasi pola migrasi. Gunakan pola IDs dan nama yang Anda tentukan dalam runbook prioritas aplikasi Anda.

Prioritas	Aturan pemetaan
1	Dengan menggunakan data pemanfaatan atau alat pemantauan, identifikasi apakah aplikasi tersebut adalah aplikasi zombie atau idle. Jika aplikasi adalah aplikasi zombie atau idle, pilih Pola 8: Pensiun aplikasi, lalu matikan server di tumpukan aplikasi.
2	Identifikasi apakah memigrasikan aplikasi ini ke cloud memberikan nilai bisnis. Aplikasi yang hanya digunakan di tempat dan tidak dibagikan di seluruh cabang atau lokasi geografis, seperti aplikasi waktu dan kehadiran, biasanya tidak perlu dimigrasikan ke cloud. Jika memigrasi aplikasi ini tidak memberikan nilai bisnis, pilih Pola 9: Simpan di tempat.
3	Identifikasi apakah sistem operasi (OS) aplikasi didukung oleh AWS, layanan AWS migrasi, vendor, atau alat migrasi rehost Anda, lalu lakukan hal berikut: • Jika OS didukung, pilih Pola 1: Rehost ke Amazon EC2 menggunakan Layanan Migrasi Aplikasi atau Pabrik Migrasi Cloud. • Jika OS tidak didukung, pilih Pola 3: Replatform ke Amazon CloudFormation EC2 menggunakan.
4	Identifikasi apakah aplikasi memiliki versi perangkat lunak sebagai layanan (SaaS) atau setara, dan kemudian evaluasi manfaat dan biaya pindah ke platform baru ini. Jika kriteria ini terpenuhi, pilih Pola 10: Pembelian kembali dan tingkatkan ke SaaS.

Prioritas	Aturan pemetaan
5	Identifikasi apakah database lokal aplikasi dapat dimigrasi ke layanan homogen di cloud, seperti memigrasikan database Oracle lokal ke Amazon RDS for Oracle atau memigrasi kan database MySQL lokal ke Amazon Aurora MySQL database Edition yang kompatibe l dengan Amazon Aurora. Jika kriteria ini terpenuhi, gunakan Pola 2: Replatform ke Amazon RDS menggunakan AWS DMS dan. AWS SCT
6	Identifikasi apakah aplikasi menggunakan Microsoft .NET Core (.NET 5 atau .NET 6), Java, PHP, atau bahasa pemrograman open-source lainnya dan apakah aplikasi tersebut di-host di Microsoft Windows Server. Evaluasi apakah biaya replatforming dapat dibenarkan. Jika kriteria ini terpenuhi, pilih Pola 7: Replatfor m dari Windows ke Linux di Amazon EC2.
7	Identifikasi penyimpanan file lokal dan bersama lokal yang bergantung pada aplikasi Anda, lalu tentukan apakah harus disertakan dalam migrasi. Jika penyimpanan file lokal dan bersama harus dimigrasikan, pilih Pola 4: Platform Ulang ke Amazon EFS menggunakan AWS DataSync atau. AWS Transfer Family

Prioritas	Aturan pemetaan
8	Identifikasi apakah server aplikasi adalah server mainframe atau midrange, seperti IBM AS/400 atau Apache Spark, dan konfirmasikan bahwa aplikasi tersebut kompatibel dengan emulator. Jika kriteria ini terpenuhi, gunakan Pattern 6: Replatform mainframe atau server midrange ke Amazon EC2 menggunakan emulator.
9	Identifikasi apakah ini adalah aplikasi berbasis warisan, monolitik, atau mainframe yang tidak dapat lagi memenuhi permintaan bisnis karena keterbatasannya. Misalnya, identifikasi apakah aplikasi dapat menskalakan, berintegrasi dengan aplikasi terkait, atau mahal dan sulit dirawat. Jika aplikasi memenuhi salah satu kriteria ini, pilih Pola 11: Arsitek ulang aplikasi.

Tentukan proses pemetaan aplikasi

Saat Anda memetakan aplikasi ke pola migrasi, akan sangat membantu jika Anda meminta data penemuan untuk aplikasi dari tim penemuan. Data ini biasanya mencakup informasi seperti pola migrasi yang direkomendasikan (kadang-kadang disebut pola R atau skor R), informasi pemanfaatan, dependensi aplikasi, dan informasi lain yang dapat Anda gunakan dalam penilaian. Saat Anda menjelajahi aplikasi ini secara rinci, Anda mungkin memutuskan untuk mengubah pola migrasi yang sebelumnya diidentifikasi.

Ketika Anda memiliki data, Anda dapat membandingkan aplikasi dengan kriteria bisnis dan teknis yang Anda identifikasiLangkah 2: Identifikasi driver bisnis dan teknis. Anda merekam driver Anda di runbook prioritas aplikasi Anda. Mengevaluasi aplikasi berdasarkan kriteria dapat mengarahkan Anda untuk memilih beberapa pola migrasi untuk aplikasi, atau mungkin mengarahkan Anda untuk menghilangkan pola berdasarkan biaya, jadwal, atau batasan lainnya.

Berikut ini adalah contoh driver bisnis yang mengharuskan Anda untuk menggunakan beberapa pola migrasi pada satu aplikasi. Anda ingin memigrasikan database SQL Server lokal ke Amazon

DynamoDB, tetapi karena kontrak untuk pusat data kedaluwarsa, bisnis ingin memigrasikannya lebih cepat daripada jadwal yang diusulkan untuk memplatform ulang. Untuk mengatasi driver bisnis ini, Anda merevisi rencana migrasi untuk aplikasi menjadi pendekatan dua pola. Pertama, Anda meng-host ulang aplikasi ke cloud untuk menghapusnya dari pusat data. Kemudian, setelah aplikasi berada di cloud, Anda dapat memplatform ulang sesuai dengan jadwal yang diusulkan.

Anda juga harus mempertimbangkan apakah aplikasi tersebut adalah aplikasi n-tier, yang merupakan aplikasi yang terdiri dari beberapa tingkatan. Tingkat aplikasi adalah sekelompok server fisik yang meng-host lapisan horizontal aplikasi Anda. Aplikasi N-tier lebih kompleks karena setiap tingkatan mungkin memerlukan strategi yang berbeda, dan Anda mungkin memilih untuk memigrasikan tingkatan aplikasi dalam gelombang yang berbeda. Misalnya, jika Anda memiliki aplikasi yang terdiri dari presentasi, layanan bisnis, dan tingkatan basis data, ada kemungkinan Anda dapat memetakan pola yang berbeda untuk setiap tingkatan.

Anda kemudian mengevaluasi aplikasi terhadap aturan pemetaan aplikasi Anda, yang Anda tentukan dalam runbook prioritas aplikasi Anda. Untuk informasi lebih lanjut, lihat Aturan pemetaan aplikasi sebelumnya di bagian ini.

Setelah Anda memetakan aplikasi Anda ke satu atau beberapa pola, tinjau dan verifikasi keputusan ini dengan pemilik aplikasi. Pemilik aplikasi harus mengonfirmasi pola yang dipilih, dan mereka akan membantu Anda merencanakan dan melakukan migrasi. Pada saat ini, pemilik aplikasi mungkin juga memberikan wawasan berdasarkan pengalaman mereka atau berbagi masalah apa pun yang mereka antisipasi dengan migrasi.

Ketika Anda telah memetakan aplikasi ke satu atau beberapa pola migrasi dan mengonfirmasi pola dengan pemilik aplikasi, Anda merekam aplikasi, ID pola, nama pola, dan driver yang relevan dalam tabel pemetaan aplikasi di runbook prioritas aplikasi Anda.

Dalam template playbook portofolio, template Runbook untuk prioritas aplikasi mencakup step-by-step proses standar untuk pemetaan aplikasi. Tentukan proses Anda sebagai berikut:

1. Buka runbook prioritas aplikasi Anda.
2. Di bagian Proses lokakarya Aplikasi, modifikasi proses standar untuk memenuhi kebutuhan kasus penggunaan Anda.
3. Simpan runbook prioritas aplikasi.
4. Pertahankan proses dalam runbook prioritas aplikasi.

Tabel berikut adalah contoh tabel pemetaan aplikasi. Template Runbook yang disediakan untuk prioritas aplikasi menyertakan tabel kosong tempat Anda dapat merekam hasil dari proses pemetaan aplikasi.

> **Note**
>
> Pola IDs dan nama dalam tabel ini sesuai dengan pola sampel di Langkah 4: Validasi pola migrasi. Gunakan pola IDs dan nama yang Anda tentukan dalam runbook prioritas aplikasi Anda.

Nama aplikasi	ID Pola	Nama pola	Driver bisnis dan teknis ditangani
Situs web perusahaan	1	Rehost ke Amazon EC2 menggunakan Layanan Migrasi Aplikasi atau Pabrik Migrasi Cloud	• Pintu keluar pusat data • Mengurangi biaya operasional
Sistem SDM	8	Pensiun aplikasi	• Mengurangi biaya operasional
Aplikasi waktu dan kehadiran	9	Pertahankan di tempat	• Mengurangi biaya operasional • Mengurangi risiko dan dampak
Sistem PO	3	Replatform ke Amazon EC2 menggunakan CloudFormation	• Integrasi teknologi • Batasan penyimpanan dan komputasi • end-of-lifeDukungan perangkat keras dan perangkat lunak

Nama aplikasi	ID Pola	Nama pola	Driver bisnis dan teknis ditangani
			• Meningkatkan keamanan dan kepatuhan
Sistem CRM	10	Beli kembali dan tingkatkan ke SaaS	• Mengurangi biaya operasional • Integrasi teknologi • end-of-lifeDukungan perangkat keras dan perangkat lunak • Mempercepat pengembangan, inovasi, dan pertumbuhan
Sistem aset tetap	7	Replatform dari Windows ke Linux di Amazon EC2	• Mengurangi biaya operasional
Penyimpanan file ERP	4	Platform ulang ke Amazon EFS menggunakan AWS DataSync atau AWS Transfer Family	• Batasan penyimpanan dan komputasi

Nama aplikasi	ID Pola	Nama pola	Driver bisnis dan teknis ditangani
Sistem buku besar	6	Rehost server mainframe atau midrange ke Amazon EC2 menggunakan emulator	• Pintu keluar pusat data • Integrasi teknologi • Meningkatkan keamanan dan kepatuhan • end-of-lifeDukungan perangkat keras dan perangkat lunak • Batasan penyimpanan dan komputasi • Memodernisasi arsitektur aplikasi

Nama aplikasi	ID Pola	Nama pola	Driver bisnis dan teknis ditangani
Buku besar	11	Arsitek ulang aplikasi	• Mengurangi biaya operasional • Integrasi teknologi • Meningkatkan keamanan dan kepatuhan • end-of-lifeDukungan perangkat keras dan perangkat lunak • Batasan penyimpanan dan komputasi • Memodernisasi arsitektur aplikasi • Skalabilitas dan ketahanan • Mempercepat pengembangan, inovasi, dan pertumbuhan

Kriteria keluar langkah

- Anda telah mendokumentasikan hal-hal berikut dalam runbook prioritas aplikasi Anda:
  - Aturan pemetaan aplikasi
  - Proses pemetaan aplikasi
- Anda telah memvalidasi aturan dan proses pemetaan dengan menggunakan satu atau beberapa aplikasi proof-of-concept (POC).

Langkah 3: (Opsional) Tentukan status target aplikasi

Pada langkah ini, Anda menentukan atribut dan proses yang Anda gunakan untuk mendokumentasikan status target, atau status calon, untuk aplikasi. Status target adalah bagaimana aplikasi beroperasi di lingkungan cloud target setelah migrasi. Lingkungan target bervariasi berdasarkan platform target atau layanan dan persyaratan bisnis Anda. Lingkungan target bisa menjadi AWS Cloud or AWS Managed Services (AMS).

Mendefinisikan status target membantu manajer proyek, konsultan migrasi, arsitek, pemilik aplikasi, dan pemangku kepentingan berkolaborasi secara efektif. Dengan menggunakan proses ini, tim dapat mengidentifikasi dan menyelesaikan masalah terlebih dahulu dan lebih efisien menerapkan lingkungan status target.

Untuk beberapa aplikasi, langkah ini opsional. Anda dapat melewati langkah ini jika aplikasi yang Anda migrasi mandiri atau kompleksitas rendah. Strategi migrasi yang tidak mengubah aplikasi, seperti rehost, mungkin tidak memerlukan langkah ini. Namun, untuk strategi migrasi yang lebih kompleks, seperti replatform dan arsitek ulang, Anda harus menentukan status target sebelum memulai migrasi.

Lokakarya ini memberi Anda pemahaman mendalam tentang keadaan aplikasi saat ini, jadi ada baiknya untuk menyusun status target setelah menyelesaikan lokakarya. Selain itu, pemetaan aplikasi ke pola migrasi memberikan wawasan tambahan dan membantu Anda mengidentifikasi apakah menentukan status target diperlukan. Misalnya, jika Anda memetakan aplikasi ke pola Rehost ke Amazon EC2 menggunakan Layanan Migrasi Aplikasi atau Pabrik Migrasi Cloud, Anda telah mengidentifikasi bahwa strateginya adalah rehost, dan Anda mungkin tidak perlu menentukan status target untuk aplikasi ini.

Atribut status target

Saat menentukan status target dan membuat keputusan tentang aplikasi, kami sarankan Anda mempertimbangkan atribut status target berikut:

- AWS Well-Architected Tool— Tinjau status target aplikasi terhadap AWS Well-Architected Framework untuk membantu meningkatkan keamanan, kinerja, dan ketahanan aplikasi di cloud.
- Target landing zone — Biasanya, pada akhir fase mobilisasi, Anda seharusnya telah membangun landing zone lengkap yang siap menjalankan aplikasi pilot. Landing zone harus sudah dikonfigurasi dengan arsitektur multi-akun, identitas dan manajemen akses, tata kelola, keamanan data, desain jaringan, dan logging. Anda menggunakan aplikasi pilot untuk memverifikasi bahwa landing zone sudah lengkap. Verifikasi bahwa Anda dapat meluncurkan dan menjalankan aplikasi pilot Anda di

target landing zone yang ada. Jika Anda perlu memodifikasi landing zone untuk aplikasi, beri tahu tim landing zone tentang kebutuhan Anda. Misalnya, aplikasi Anda mungkin memerlukan akses ke layanan yang di-host di akun terpisah, atau aplikasi Anda mungkin memerlukan perutean khusus ke virtual private cloud (VPC).

- Dependensi — Identifikasi aplikasi apa pun yang diandalkan aplikasi Anda agar berfungsi dengan baik. Misalnya, aplikasi Anda mungkin bergantung pada database, penyimpanan, atau layanan pihak ketiga, seperti gateway pembayaran atau layanan web eksternal, atau aplikasi Anda mungkin bergantung pada aplikasi lain di lingkungan Anda. Untuk mengakses dependensi ini, Anda mungkin memerlukan perutean atau konfigurasi khusus, seperti menghubungkan ke layanan direktori untuk otentikasi.
- Aplikasi dependen — Identifikasi aplikasi apa pun yang bergantung pada aplikasi Anda agar berfungsi dengan baik. Anda mungkin perlu mengkonfigurasi ulang dan memperbarui aplikasi ini untuk mencegah waktu henti selama migrasi.
- Keamanan dan kepatuhan - Tinjau lingkungan target dengan tim keamanan dan kepatuhan, dan identifikasi celah apa pun. Aplikasi ini mungkin terdiri dari beberapa komponen, lapisan logis, atau beberapa tingkatan. Bergantung pada persyaratan kepatuhan, Anda mungkin tidak dapat memigrasikan beberapa komponen ini ke lingkungan target, atau Anda mungkin memerlukan langkah-langkah keamanan tambahan saat memigrasikan beban kerja. Persyaratan keamanan dan kepatuhan yang umum adalah residensi data, enkripsi dalam perjalanan, dan enkripsi saat istirahat. Ini memerlukan konfigurasi tambahan di lingkungan target Anda. Misalnya, Anda mungkin perlu mengonfigurasi sertifikat untuk mengamankan komunikasi, atau Anda mungkin memerlukan kunci enkripsi untuk mengamankan data saat istirahat. Anda mungkin juga perlu memilih beberapa pola migrasi untuk aplikasi sehingga beberapa tingkatan aplikasi tetap berada di lokasi dan tingkatan lainnya dimigrasikan ke cloud.
- Dependensi penyimpanan — Tinjau dependensi penyimpanan aplikasi Anda dan tentukan bagaimana migrasi aplikasi ke lingkungan target akan memengaruhi dependensi ini. Misalnya, jika aplikasi bergantung pada penyimpanan jaringan, seperti penyimpanan terpasang jaringan (NAS) atau jaringan area penyimpanan (SAN), Anda perlu merencanakan layanan penyimpanan di cloud, seperti Amazon Simple Storage Service (Amazon S3) atau Amazon. FSx Anda juga perlu merencanakan bagaimana Anda akan memigrasikan data Anda ke lingkungan cloud target agar aplikasi Anda tetap berjalan.
- Database — Tinjau strategi migrasi untuk database apa pun yang digunakan aplikasi. Apakah Anda akan memplatform ulang ke layanan database baru, seperti Amazon Relational Database Service (Amazon RDS), Amazon Aurora, atau Amazon DynamoDB? Apakah Anda akan meng-host ulang database Anda di lingkungan target? Dalam beberapa kasus, terutama untuk database

monolitik, Anda perlu memfaktorkan ulang database untuk memenuhi persyaratan teknis, seperti latensi sub-detik, atau untuk memanfaatkan fitur dari jenis database tertentu. AWS Seperti persyaratan kepatuhan residensi data, Anda perlu mengidentifikasi data mana yang harus disimpan di tempat dan mana yang harus dimigrasikan ke cloud. Misalnya, Anda mungkin perlu menyimpan tabel database lokal untuk informasi pelanggan, dan Anda dapat memigrasikan sisa database ke cloud.

- Komponen aplikasi — Tinjau komponen yang bergantung pada aplikasi Anda. Tentukan apakah aplikasi Anda bergantung pada komponen yang tidak didukung oleh lingkungan target. Jika lingkungan target tidak mendukung semua komponen aplikasi, Anda perlu memfaktorkan ulang aplikasi untuk mengurangi masalah. Misalnya, jika Anda memiliki aplikasi.NET Framework yang bergantung pada komponen khusus Windows seperti Component Object Model (COM) Interop, COM+, atau registri Windows, untuk memplatform ulang aplikasi pada sistem operasi Linux, Anda harus memfaktorkan ulang aplikasi ke .NET Core.
- Tingkatan aplikasi — Identifikasi jumlah tingkatan dalam aplikasi Anda. Apakah aplikasi n-tier, two-tier, atau standalone? Konfirmasikan bahwa Anda memahami pola migrasi untuk setiap tingkatan. Misalnya, aplikasi Anda mungkin memiliki tingkat presentasi (atau web) yang menghosting antarmuka pengguna, tingkat aplikasi yang menghosting layanan bisnis, dan tingkat basis data yang menghosting database, dan setiap tingkat mungkin memerlukan pola migrasi yang berbeda.
- Pemulihan bencana - Identifikasi rencana pemulihan bencana (DR) aplikasi saat ini dan masa depan, termasuk tujuan titik pemulihan (RPO) dan tujuan waktu pemulihan (RTO). Putuskan apakah akan menggunakan rencana DR lokal yang ada atau untuk menjelajahi strategi DR baru di cloud. Untuk informasi selengkapnya, lihat bagian Opsi pemulihan bencana di cloud dan tujuan Pemulihan (RTO dan RPO) di Disaster Recovery of Workloads on AWS: Recovery in the Cloud whitepaper.

Tentukan proses status target

Untuk menentukan status target aplikasi, kami sarankan Anda menggunakan templat yang disediakan, Lembar kerja status target aplikasi (format Excel), yang tersedia di templat buku pedoman portofolio. Template mencakup atribut standar yang dapat Anda gunakan atau modifikasi. Tentukan proses untuk mendokumentasikan status target aplikasi sebagai berikut:

1. Buka lembar kerja status target aplikasi.
2. Tinjau atribut default dan buat perubahan apa pun untuk kasus penggunaan Anda.
3. Simpan lembar kerja.
4. Buka runbook prioritas aplikasi Anda.

5. Di bagian Status aplikasi Target, lakukan hal berikut:
   a. Di bagian Kapan menyelesaikan proses ini, tetapkan kriteria yang memungkinkan tim portofolio untuk menentukan apakah mereka perlu menentukan status target aplikasi.
   b. Perbarui bagian atribut sesuai kebutuhan.
   c. Perbarui bagian proses sesuai kebutuhan untuk kasus penggunaan Anda.
6. Simpan runbook prioritas aplikasi.

Sampel dari status target aplikasi

Tabel berikut menunjukkan contoh bagaimana Anda menggunakan lembar kerja status target Aplikasi untuk mendokumentasikan status target aplikasi.

Aplikasi	Contoh
Platform target	AWS Cloud
Zona pendaratan	Memerlukan akses ke layanan direktori lokal Membutuhkan AWS Control Tower untuk memusatkan pengelolaan beberapa akun dan layanan di seluruh organisasi
Dependensi	Active Directory, gateway pembayaran, sistem inventaris
Aplikasi tergantung	Tidak ada
Keamanan	Enkripsi saat istirahat dan dalam transit
Kepatuhan	PCI, SOC
Dependensi penyimpanan	Boot drive terpasang, NAS, berbagi jaringan
Basis Data	Saat ini: Oracle DB Cloud: Amazon RDS for Oracle
Komponen aplikasi	.NET Framework 4.5

Aplikasi	Contoh
Tingkatan aplikasi	N-tingkat Front-end, layanan bisnis, layanan data dan agen, database
Pemulihan bencana	RPO: 1 menit, RTO: 5 menit Strategi DR saat ini adalah siaga hangat DR di wilayah AS mana pun

Kriteria keluar langkah

- Di lembar kerja status target aplikasi, Anda telah menentukan atribut yang ingin Anda sertakan dalam proses status target.
- Dalam runbook prioritas aplikasi Anda, Anda telah melakukan hal berikut:
  - Anda telah menetapkan kriteria kapan tim portofolio diharapkan untuk menentukan status target aplikasi.
  - Anda telah memperbarui proses untuk menentukan status target berdasarkan kasus penggunaan Anda.

Langkah 4: Selesaikan proses penyelaman mendalam aplikasi

Sekarang, Anda menentukan bagaimana alur kerja portofolio menggunakan lokakarya, aturan, dan proses yang Anda buat dalam tugas ini untuk melakukan penyelaman mendalam ke dalam aplikasi. Ini adalah proses yang direferensikan oleh alur kerja portofolio dalam tahap implementasi migrasi.

Sesuaikan proses ini di runbook prioritas aplikasi sebagai berikut:

1. Buka runbook prioritas aplikasi Anda.
2. Di Tahap 2: Lakukan aplikasi deep dive, modifikasi proses yang sesuai untuk kasus penggunaan dan lingkungan Anda.
3. Simpan runbook prioritas aplikasi.
4. Bagikan runbook prioritas aplikasi Anda dengan tim untuk ditinjau.

Tugas 5: Mendefinisikan proses perencanaan gelombang

Perencanaan gelombang adalah tonggak penting dalam migrasi besar. Dalam rencana gelombang, Anda mengelompokkan aplikasi serupa bersama-sama, memperhitungkan infrastruktur dan dependensi aplikasi (seperti database bersama), prioritas aplikasi, kesamaan arsitektur aplikasi, dan fungsionalitas bisnis. Anda kemudian meninjau rencana gelombang dengan tim aplikasi dan infrastruktur untuk mengonfirmasi ketersediaannya selama migrasi dan jendela cutover yang ditentukan.

Berdasarkan penerapan kehidupan nyata untuk berbagai AWS pelanggan, berikut ini adalah beberapa praktik terbaik untuk perencanaan gelombang:

- Rencanakan gelombang migrasi setidaknya 4-5 gelombang sebelumnya. Ini membantu memastikan bahwa selalu ada cukup server untuk alur kerja migrasi.
- Gagal cepat. Anda harus mulai dengan beberapa aplikasi dengan kompleksitas rendah dan menerapkan pembelajaran Anda ke gelombang selanjutnya.
- Pada gelombang awal (gelombang 1-5), pilih lebih sedikit server (kurang dari 10), aplikasi dengan kompleksitas rendah, dan aplikasi di lingkungan yang lebih rendah, seperti lingkungan pengembangan atau pengujian. Secara bertahap perkenalkan lebih banyak kompleksitas dan lebih banyak server ke dalam gelombang saat Anda maju.
- Perencanaan gelombang adalah proses yang berkelanjutan, bukan tugas satu kali. Jangan mencoba merencanakan semua gelombang sekaligus.
- Jika Anda menggunakan alat penemuan portofolio dan memiliki fitur penilaian kompleksitas, gunakan dalam perencanaan gelombang. Migrasikan aplikasi dengan kompleksitas terendah terlebih dahulu.

Tugas ini terdiri dari langkah-langkah berikut:

- Langkah 1: Tentukan proses grup bergerak
- Langkah 2: Tentukan kriteria pemilihan perencanaan gelombang
- Langkah 3: Selesaikan proses perencanaan gelombang

Langkah 1: Tentukan proses grup bergerak

Pada langkah ini, Anda mengidentifikasi application-to-server dependensi apa pun dan menentukan aturan yang akan digunakan untuk menentukan server mana yang harus dipindahkan bersama,

sebagai grup bergerak. Grup bergerak adalah blok server atau aplikasi yang harus dipindahkan bersama dalam grup. Ini adalah blok bangunan gelombang migrasi, di mana setiap gelombang terdiri dari satu atau lebih kelompok bergerak, tergantung pada jumlah server di setiap grup bergerak.

Identifikasi dependensi aplikasi

Berikut ini adalah pertimbangan utama saat mengelompokkan aplikasi yang saling bergantung dalam grup bergerak:

- Pertimbangkan dependensi infrastruktur, seperti:
  - Sebuah aplikasi mungkin memiliki beberapa database, dan database tersebut dapat dibagi oleh aplikasi lain.
  - Aplikasi mungkin tergantung pada aplikasi lain.
  - Server mungkin meng-host database untuk beberapa aplikasi.
- Pertimbangkan dependensi bisnis dan operasional, seperti:
  - Karena dampak bisnis atau jadwal operasional (seperti backup atau patching), aplikasi hanya dapat dimigrasikan selama jendela tertentu.
  - Pemilik aplikasi hanya tersedia untuk satu jendela cutover migrasi, jadi semua aplikasi pemilik harus berada dalam grup pemindahan yang sama.

Anda mengidentifikasi dependensi infrastruktur dalam proses lokakarya aplikasi atau ketika Anda menentukan status target. Anda dapat mengidentifikasi dependensi infrastruktur melalui proses otomatis atau manual. Untuk mengotomatiskan identifikasi dependensi infrastruktur, Anda dapat menggunakan alat penemuan, seperti Flexera One Cloud Migration and Modernization atau TDS. TransitionManager Untuk proses manual, validasi informasi CMDB dengan tim aplikasi dan infrastruktur.

Anda mengidentifikasi dependensi bisnis dan operasional dalam proses lokakarya aplikasi.

Sebagai titik awal untuk membangun runbook perencanaan gelombang Anda sendiri, kami sarankan Anda menggunakan template Runbook untuk perencanaan gelombang (format Microsoft Word) yang disertakan dalam templat buku pedoman portofolio. Dokumentasikan dependensi untuk migrasi Anda sebagai berikut:

1. Buka runbook perencanaan gelombang Anda.
2. Di bagian Application dependencies, catat dependensi. Identifikasi jenis (infrastruktur, bisnis, atau operasional), ketergantungan, dan deskripsi singkat tentang ketergantungan.

3. Simpan runbook perencanaan gelombang.
4. Pertahankan dependensi dalam runbook perencanaan gelombang. Saat Anda maju, Anda mungkin mengidentifikasi dependensi baru.

Tabel berikut menunjukkan contoh dependensi.

Tipe	Dependensi	Deskripsi
Infrastruktur	Basis Data	Database dibagikan dengan aplikasi lain
Infrastruktur	Toko file	Aplikasi ini menggunakan penyimpanan file pusat yang dapat dipisahkan, atau semua aplikasi terkait harus bermigrasi bersama
Infrastruktur	Aplikasi	Aplikasi tergantung pada satu atau lebih aplikasi lain untuk berfungsi, seperti mengekstr ak, mengubah, dan memuat (ETL) pekerjaan
Bisnis	Pemadaman bisnis	Jendela pemadaman khusus dan disetujui untuk aplikasi
Operasional	Jendela tambalan	Tugas operasional terjadwal, seperti menambal, yang dapat memengaruhi pemotongan migrasi

Tentukan aturan grup pindah

Setelah merekam dependensi di runbook perencanaan gelombang, Anda harus membuat aturan grup pemindahan berdasarkan dependensi tersebut. Aturan-aturan ini mengatur bagaimana server dikelompokkan bersama ke dalam grup bergerak. Gunakan langkah-langkah berikut untuk membuat aturan Anda:

1. Tinjau dependensi yang Anda tentukan di bagian sebelumnya.
2. Pilih dependensi yang memengaruhi apakah aplikasi harus dipindahkan bersama, dalam grup pindahan. Tidak semua dependensi memerlukan aplikasi untuk dimigrasikan bersama. Misalnya, ketergantungan infrastruktur pada Microsoft Active Directory tidak boleh dipertimbangkan ketika mendefinisikan grup pindah karena merupakan ketergantungan umum untuk semua aplikasi. Anda harus membangun pengontrol domain di cloud sebelum memigrasikan aplikasi apa pun.
3. Konversikan dependensi yang mengharuskan aplikasi dipindahkan bersama ke aturan grup pemindahan.

Jika aplikasi cocok dengan salah satu aturan, maka semua server terkait harus ditempatkan dalam grup pemindahan yang sama sehingga mereka dimigrasikan bersama.

Dokumentasikan aturan grup pemindahan untuk migrasi Anda sebagai berikut:

1. Buka runbook perencanaan gelombang Anda.
2. Di bagian Memindahkan aturan grup, catat aturan grup pemindahan sesuai urutan prioritas.
3. Simpan runbook perencanaan gelombang.
4. Pertahankan aturan dalam runbook perencanaan gelombang. Saat Anda maju, Anda mungkin mengidentifikasi aturan baru.

Tabel berikut menunjukkan contoh memindahkan aturan grup.

Aturan	Pindahkan aturan grup
1	Aplikasi dengan database bersama harus bermigrasi bersama.
2	Aplikasi yang memiliki pemilik aplikasi yang sama harus bermigrasi bersama.
3	Aplikasi dengan jendela patch yang sama harus bermigrasi bersama.

Langkah 2: Tentukan kriteria pemilihan perencanaan gelombang

Setelah Anda membuat grup bergerak, Anda perlu mengumpulkan kelompok bergerak serupa bersama-sama untuk membentuk gelombang migrasi. Pada langkah ini, Anda menentukan kriteria yang Anda gunakan untuk memilih satu atau beberapa grup bergerak untuk setiap gelombang.

Memahami ukuran setiap kelompok bergerak sangat penting untuk perencanaan gelombang yang sukses. Tujuannya adalah untuk mengukur setiap gelombang sehingga migrasi tetap gesit dan mempertahankan jaringan server yang sehat. Gelombang yang terlalu besar bisa sulit untuk beradaptasi dengan perubahan dalam rencana migrasi, dan gelombang yang terlalu kecil mungkin tidak menyediakan server yang cukup untuk mencapai kecepatan migrasi yang diinginkan.

Kami menyarankan Anda mempertimbangkan kriteria berikut saat mengukur gelombang:

- Gelombang pertama kecil — Gelombang awal harus lebih kecil, dengan kurang dari 10 server, dan kemudian Anda dapat secara bertahap meningkatkan jumlah server di setiap gelombang. Ini memungkinkan Anda untuk gagal dengan cepat dan membangun pelajaran yang dipetik. Misalnya, memigrasikan aplikasi dengan 3 server sebelum memigrasikan aplikasi dengan 20 server.
- Sumber Daya — Identifikasi berapa banyak server yang dapat dimigrasi oleh tim migrasi dalam satu gelombang. Ukuran standar adalah bahwa tim migrasi yang terdiri dari empat arsitek dapat bermigrasi hingga 50 server dalam seminggu untuk pola rehost. Gabungkan grup pindahan untuk membentuk gelombang migrasi yang tidak melebihi kapasitas tim migrasi.
- Kelincahan — Gelombang harus adaptif terhadap setiap perubahan dalam rencana migrasi. Jika Anda harus menjadwal ulang server, Anda harus dapat menjadwal ulang seluruh grup pemindahan server yang terpengaruh.
- Ukuran penyimpanan — Migrasikan aplikasi yang lebih kecil terlebih dahulu. Misalnya, memigrasikan aplikasi 100 GB sebelum aplikasi 2 TB.
- Lingkungan aplikasi — Migrasikan aplikasi di lingkungan yang lebih rendah, seperti lingkungan pengembangan atau pengujian, sebelum aplikasi di lingkungan produksi.
- Kompleksitas aplikasi — Migrasikan aplikasi yang kurang kompleks dengan dependensi eksternal yang lebih sedikit terlebih dahulu.
- Kekritisan aplikasi — Migrasikan aplikasi non-kritis sebelum aplikasi mission-critical.
- Basis pengguna — Migrasikan aplikasi yang memiliki basis pengguna kecil terlebih dahulu. Misalnya, memigrasikan aplikasi yang memiliki 10 pengguna sebelum aplikasi yang memiliki 10.000 pengguna.

- Bandwidth jaringan — Ukuran gelombang tidak boleh melebihi bandwidth jaringan. Untuk informasi selengkapnya, lihat prinsip migrasi Anda, yang Anda tetapkan sesuai dengan petunjuk di buku pedoman Foundation untuk migrasi AWS besar.

Dokumentasikan kriteria seleksi untuk perencanaan gelombang sebagai berikut:

1. Buka runbook perencanaan gelombang Anda.
2. Di bagian Kriteria pemilihan perencanaan gelombang, catat kriteria yang ingin Anda gunakan untuk migrasi Anda.
3. Simpan runbook perencanaan gelombang.
4. Pertahankan kriteria dalam runbook perencanaan gelombang. Saat Anda maju, Anda mungkin perlu menyesuaikan kriteria atau menambahkan kriteria baru.

Tabel berikut menunjukkan contoh kriteria pemilihan perencanaan gelombang.

Kriteria	Deskripsi
Identifikasi aplikasi yang paling tidak kompleks	Identifikasi aplikasi dengan skor kompleksitas yang lebih tinggi dalam kelompok bergerak.
Lingkungan yang lebih rendah terlebih dahulu	Aplikasi non-kritis dalam lingkungan yang lebih rendah, seperti lingkungan pengembangan atau pengujian, harus bermigrasi terlebih dahulu. Aplikasi penting dalam lingkungan produksi, seperti yang menghasilkan pendapatan, harus bermigrasi terakhir.
Gagal cepat	Bentuk gelombang awal dengan kurang dari 10 server.
Kekuatan tim migrasi	Identifikasi berapa banyak server yang dapat dipotong oleh setiap tim migrasi.
Gabungkan grup bergerak serupa	Gabungkan kelompok bergerak berdasarkan kesamaan. Misalnya, grup pemindahan mungkin berbagi pemilik aplikasi, pusat data sumber, atau AWS akun target yang sama.

Kriteria	Deskripsi
Ukuran gelombang	Gelombang tidak boleh melebihi 50 server secara total.

Kriteria keluar langkah

- Anda telah mengidentifikasi kriteria perencanaan gelombang untuk kasus penggunaan Anda dan mendokumentasikannya di buku runbook perencanaan gelombang Anda.

Langkah 3: Selesaikan proses perencanaan gelombang

Sekarang setelah Anda menentukan cara membuat grup bergerak dan menetapkan kriteria yang Anda gunakan untuk menggabungkan grup pindah ke gelombang migrasi, Anda harus menentukan proses untuk merencanakan gelombang. Pada langkah ini, Anda memperbarui runbook perencanaan gelombang Anda untuk merekam proses perencanaan gelombang lengkap, dan Anda mengonfirmasi bahwa Anda memiliki alat dasbor yang dapat digunakan tim untuk merekam informasi gelombang.

Pada langkah ini, kami menyarankan Anda menggunakan template Dasbor yang disediakan untuk perencanaan gelombang dan migrasi, yang tersedia di templat buku pedoman portofolio. Template ini dirancang untuk membantu tim portofolio dan berfungsi sebagai titik awal untuk menyusun data, membantu menganalisis portofolio aplikasi, mengidentifikasi application-to-server dependensi, dan akhirnya merencanakan gelombang migrasi. Anda dapat memodifikasi template ini sesuai kebutuhan untuk lingkungan Anda.

Dokumentasikan proses perencanaan gelombang sebagai berikut:

1. Buka template Dasbor untuk perencanaan gelombang dan migrasi.
2. Ubah dasbor sesuai kebutuhan untuk kasus penggunaan Anda. Misalnya, Anda dapat menambahkan lembar kerja untuk mengekstrak inventaris server, menambahkan tabel pivot atau bagan baru, atau mengimpor informasi sumber dengan menggunakan fungsi. `VLOOKUP`
3. Simpan template dasbor.
4. Buka runbook perencanaan gelombang Anda.
5. Di bagian Tahap 2: Lakukan perencanaan gelombang, modifikasi proses standar yang disediakan untuk memenuhi kebutuhan kasus penggunaan Anda.
6. Simpan runbook perencanaan gelombang.

7. Bagikan runbook perencanaan gelombang Anda dengan tim untuk ditinjau.
8. Pertahankan proses di runbook perencanaan gelombang. Proses ini bertindak sebagai prosedur operasi standar untuk merencanakan gelombang untuk migrasi besar Anda.

Kriteria keluar tugas

- Anda telah mendokumentasikan hal-hal berikut di runbook perencanaan gelombang Anda:
  - Dependensi aplikasi
  - Aturan grup pemindahan aplikasi, tercantum dalam urutan prioritas
  - Kriteria pemilihan perencanaan gelombang
  - Proses perencanaan gelombang

Tahap 2: Menerapkan migrasi besar

Pada tahap 1, menginisialisasi migrasi besar, Anda menentukan penilaian portofolio dan proses perencanaan gelombang dan mendokumentasikannya dalam runbook. Pada tahap 2, menerapkan migrasi besar, Anda menyelesaikan proses ini dan mengulanginya untuk setiap sprint hingga migrasi selesai.

Tim portofolio menyelesaikan tugas penilaian portofolio dan perencanaan gelombang berikut di tahap 2:

- Tugas 1: Memprioritaskan aplikasi
- Tugas 2: Melakukan aplikasi deep dive
- Tugas 3: Melakukan perencanaan gelombang dan pengumpulan metadata

> **Note**
>
> Penilaian portofolio dan perencanaan gelombang bukanlah tugas satu kali. Ini adalah tugas berkelanjutan yang mendukung migrasi. Anda mengulangi semua tugas di tahap ini berkali-kali hingga migrasi selesai.

Penilaian portofolio dan proses perencanaan gelombang biasanya membutuhkan 1-2 minggu untuk setiap gelombang. Alur kerja portofolio biasanya merencanakan 4-5 gelombang sebelumnya untuk mempertahankan jaringan server yang sehat untuk alur kerja migrasi. Alur kerja portofolio mulai merencanakan gelombang pada akhir tahap inisialisasi (tahap 1), dan tahap implementasi (tahap 2) dimulai ketika alur kerja migrasi mulai memigrasi gelombang pertama aplikasi. Untuk contoh jadwal gelombang, lihat Tahap 2: Menerapkan migrasi besar di Panduan untuk migrasi AWS besar.

Melacak kemajuan

Saat Anda mulai mempersiapkan gelombang untuk migrasi, kami sarankan Anda melacak status setiap aplikasi melalui proses penilaian portofolio. Dalam template playbook portofolio, Anda dapat menggunakan template pelacakan kemajuan untuk penilaian portofolio (format Microsoft Excel). Template ini memungkinkan Anda untuk melacak hal-hal berikut untuk setiap aplikasi: skor kompleksitas, gelombang target, pemilik aplikasi, tanggal penyelesaian target untuk tugas utama

(prioritas aplikasi, penyelaman mendalam, perencanaan gelombang, dan pengumpulan data), dan kesiapan keseluruhan aplikasi untuk migrasi. Panduan dalam buku pedoman ini mencakup instruksi kapan harus memperbarui lembar pelacakan kemajuan.

Tugas 1: Memprioritaskan aplikasi

Dalam tugas ini, Anda meninjau daftar aplikasi yang tidak bermigrasi dalam portofolio Anda dan, untuk subset dari aplikasi yang tersisa, menetapkan skor kompleksitas aplikasi dan prioritas. Anda mengulangi proses ini berkali-kali di seluruh proyek migrasi.

Anda memerlukan informasi berikut untuk menyelesaikan tugas ini.

Input	Sumber
Daftar lengkap aplikasi dalam portofolio Anda yang akan Anda migrasi	Alat penemuan atau database manajemen konfigurasi (CMDB)
Strategi dan pola migrasi target, pada tingkat tinggi	Strategi migrasi dan pola Migrasi di runbook prioritas aplikasi Anda
Jumlah aplikasi yang Anda rencanakan untuk disertakan dalam gelombang	Kriteria pemilihan perencanaan gelombang di buku runbook perencanaan gelombang Anda

Ikuti petunjuk di runbook prioritas aplikasi Anda, di bagian Tahap 2: Prioritaskan aplikasi. Anda mendefinisikan proses ini di buku pedoman ini, di[Langkah 3: Selesaikan proses prioritas aplikasi](#).

Di akhir tugas ini, Anda telah menyelesaikan yang berikut ini.

Output	Deskripsi
Daftar aplikasi yang diprioritaskan	Anda telah memprioritaskan 2-3 kali jumlah aplikasi yang Anda rencanakan untuk disertakan dalam gelombang, dan Anda telah memasukkan aplikasi ini di pelacak kemajuan.

Tugas 2: Melakukan aplikasi deep dive

Dalam tugas ini, Anda melakukan penyelaman mendalam ke setiap aplikasi yang Anda prioritaskan dalam tugas sebelumnya. Ini biasanya termasuk mengirim kuesioner ke pemilik aplikasi, menganalisis dependensi aplikasi apa pun, dan menjadwalkan lokakarya aplikasi.

Anda memerlukan informasi berikut untuk menyelesaikan tugas ini.

Input	Sumber
Daftar aplikasi yang diprioritaskan	Dibuat sebelumnya dalam tahap implementasi, di Tugas 1: Memprioritaskan aplikasi
Strategi dan pola migrasi target, pada tingkat tinggi	Strategi migrasi dan pola Migrasi di runbook prioritas aplikasi

Ikuti petunjuk di runbook prioritas aplikasi Anda, di bagian Tahap 2: Lakukan penyelaman mendalam aplikasi. Anda mendefinisikan proses ini di buku pedoman ini, diLangkah 4: Selesaikan proses penyelaman mendalam aplikasi.

Di akhir tugas ini, Anda telah menyelesaikan yang berikut ini.

Output	Deskripsi
Pemetaan pola migrasi	Anda telah memetakan setiap aplikasi ke pola migrasi.
Status target aplikasi (jika ada)	Jika berlaku untuk aplikasi, Anda telah menentukan status aplikasi masa depan di cloud.

Tugas 3: Melakukan perencanaan gelombang dan pengumpulan metadata

Ini adalah tugas akhir untuk penilaian portofolio dan perencanaan gelombang. Dalam tugas ini, Anda menggunakan informasi aplikasi dan pola migrasi target untuk membangun grup pindahan,

menetapkan grup pemindahan ke gelombang, dan mengumpulkan semua metadata yang diperlukan untuk mendukung migrasi. Terakhir, Anda memberi tahu alur kerja migrasi bahwa gelombang sudah siap.

Anda memerlukan informasi berikut untuk menyelesaikan tugas ini.

Input	Sumber
Daftar aplikasi yang diprioritaskan	Dibuat sebelumnya dalam tahap implementasi, di Tugas 1: Memprioritaskan aplikasi
Pemetaan pola migrasi	Dibuat sebelumnya dalam tahap implementasi, di Tugas 2: Melakukan aplikasi deep dive
Status target aplikasi (jika ada)	Juga dibuat di Tugas 2: Melakukan aplikasi deep dive

Lakukan hal-hal berikut:

1. Ikuti instruksi di runbook perencanaan gelombang Anda, di bagian Tahap 2: Lakukan perencanaan gelombang. Anda mendefinisikan proses ini di buku pedoman ini, diLangkah 3: Selesaikan proses perencanaan gelombang.
2. Ikuti petunjuk di runbook manajemen metadata Anda, di bagian Tahap 2: Kumpulkan metadata. Anda mendefinisikan proses ini di buku pedoman ini, diLangkah 3: Dokumentasikan persyaratan metadata dan proses pengumpulan di runbook.
3. Beri tahu alur kerja migrasi bahwa rencana gelombang telah selesai dan metadata sudah siap. Komunikasi ini harus mematuhi tata kelola yang Anda tetapkan per buku pedoman tata kelola Proyek untuk AWS migrasi besar.

Di akhir tugas ini, Anda telah menyelesaikan yang berikut ini.

Output	Deskripsi
Rencana gelombang	Anda telah merencanakan gelombang, mengidentifikasi server, aplikasi, dan database

Output	Deskripsi
	dalam gelombang itu, dan menentukan tanggal mulai dan tanggal dan waktu cutover.
Metadata infrastruktur sumber	Anda telah mengumpulkan metadata infrastruktur sumber, seperti nama server dan sistem operasi.
Target metadata infrastruktur	Anda telah mengumpulkan metadata infrastruktur target, seperti subnet target, grup keamanan, dan akun. AWS
Pemberitahuan selesai	Anda telah memberi tahu alur kerja migrasi bahwa rencana gelombang dan metadata sudah siap.

Tim portofolio mengulangi ketiga tugas dalam tahap ini untuk setiap sprint hingga proyek migrasi selesai.

Sumber daya

AWS migrasi besar

Untuk mengakses seri Panduan AWS Preskriptif lengkap untuk migrasi besar, lihat Migrasi besar ke. AWS Cloud

Referensi tambahan

Alat dan layanan

- AWS Solusi Pabrik Migrasi Cloud
- Layanan Migrasi Cloud Gratis di AWS
- AWS Database Migration Service
- Migrasi dengan AWS
- Migrasi dan Modernisasi Cloud Flexera One (situs web Flexera)
- TDS TransitionManager (situs web TDS)

AWS Bimbingan Preskriptif

- Mengotomatiskan migrasi server skala besar dengan Cloud Migration Factory
- Praktik terbaik untuk menilai aplikasi yang akan pensiun selama migrasi ke AWS Cloud
- Mengevaluasi kesiapan migrasi
- Memulai dengan penemuan portofolio otomatis
- Memobilisasi organisasi Anda untuk mempercepat migrasi skala besar
- Strategi migrasi untuk database relasional
- Panduan penilaian portofolio aplikasi untuk AWS Cloud migrasi

Video

- Menjalankan migrasi skala besar ke AWS (AWS re:invent 2020)
- CloudEndure Praktik terbaik Pabrik Migrasi (AWS Re: Invent 2020)

Kontributor

Individu-individu berikut berkontribusi pada dokumen ini:

- Pratik Chunawala, Arsitek Cloud Utama, Amazon Web Services
- Dwayne Bordelon, Arsitek Aplikasi Cloud Senior, Amazon Web Services
- Rodolfo Jr. Cerrada, Arsitek Aplikasi Senior, Amazon Web Services
- Wally Lu, Konsultan Utama, Amazon Web Services

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
Dihapus VMware Cloud pada AWS	Kami menghapus referensi ke VMware Cloud AWS dan memperbarui daftar strategi dan pola migrasi umum.	Juli 5, 2024
Nama AWS solusi yang diperbarui	Kami memperbarui nama AWS solusi yang direferensikan dari Pabrik CloudEndure Migrasi ke Pabrik Migrasi Cloud.	2 Mei 2022
Publikasi awal	—	28 Februari 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: Memigrasikan database Oracle lokal Anda ke Amazon Relational Database Service (Amazon RDS) untuk Oracle di. AWS Cloud
- Pembelian kembali (drop and shop) - Beralih ke produk yang berbeda, biasanya dengan beralih dari lisensi tradisional ke model SaaS. Contoh: Migrasikan sistem manajemen hubungan pelanggan (CRM) Anda ke Salesforce.com.
- Rehost (lift dan shift) — Pindahkan aplikasi ke cloud tanpa membuat perubahan apa pun untuk memanfaatkan kemampuan cloud. Contoh: Migrasikan database Oracle lokal Anda ke Oracle pada instans EC2 di. AWS Cloud
- Relokasi (hypervisor-level lift and shift) — Pindahkan infrastruktur ke cloud tanpa membeli perangkat keras baru, menulis ulang aplikasi, atau memodifikasi operasi yang ada. Anda memigrasikan server dari platform lokal ke layanan cloud untuk platform yang sama. Contoh: Migrasikan Microsoft Hyper-V aplikasi ke AWS.
- Pertahankan (kunjungi kembali) - Simpan aplikasi di lingkungan sumber Anda. Ini mungkin termasuk aplikasi yang memerlukan refactoring besar, dan Anda ingin menunda pekerjaan itu sampai nanti, dan aplikasi lama yang ingin Anda pertahankan, karena tidak ada pembenaran bisnis untuk memigrasikannya.

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

A

ABAC

Lihat kontrol akses berbasis atribut.

layanan abstrak

Lihat layanan terkelola.

ASAM

Lihat atomisitas, konsistensi, isolasi, daya tahan.

migrasi aktif-aktif

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

migrasi aktif-pasif

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

fungsi agregat

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

AI

Lihat kecerdasan buatan.

AIOps

Lihat operasi kecerdasan buatan.

anonimisasi

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

anti-pola

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

kontrol aplikasi

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

portofolio aplikasi

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

kecerdasan buatan (AI)

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

operasi kecerdasan buatan (AIOps)

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

enkripsi asimetris

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

atomisitas, konsistensi, isolasi, daya tahan (ACID)

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

kontrol akses berbasis atribut (ABAC)

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

sumber data otoritatif

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

Zona Ketersediaan

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

AWS Kerangka Adopsi Cloud (AWS CAF)

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

AWS Kerangka Kualifikasi Beban Kerja (AWS WQF)

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

B

bot buruk

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

BCP

Lihat perencanaan kontinuitas bisnis.

grafik perilaku

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

sistem big-endian

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

klasifikasi biner

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

filter mekar

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

deployment biru/hijau

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

bot

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

botnet

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

cabang

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

akses break-glass

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

strategi brownfield

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

cache penyangga

Area memori tempat data yang paling sering diakses disimpan.

kemampuan bisnis

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

perencanaan kelangsungan bisnis (BCP)

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

C

KAFE

Lihat Kerangka Adopsi AWS Cloud.

penyebaran kenari

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

CCoE

Lihat Cloud Center of Excellence.

CDC

Lihat mengubah pengambilan data.

ubah pengambilan data (CDC)

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

rekayasa kekacauan

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

CI/CD

Lihat integrasi berkelanjutan dan pengiriman berkelanjutan.

klasifikasi

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

Enkripsi sisi klien

Enkripsi data secara lokal, sebelum target Layanan AWS menerimanya.

Pusat Keunggulan Cloud (CCoE)

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

komputasi cloud

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

model operasi cloud

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

tahap adopsi cloud

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

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

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

CMDB

Lihat database manajemen konfigurasi.

repositori kode

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

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

cache dingin

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

data dingin

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

visi komputer (CV)

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

konfigurasi drift

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

database manajemen konfigurasi (CMDB)

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

paket kesesuaian

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

integrasi berkelanjutan dan pengiriman berkelanjutan (CI/CD)

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

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

CV

Lihat visi komputer.

D

data saat istirahat

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

klasifikasi data

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

penyimpangan data

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

data dalam transit

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

jala data

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

minimalisasi data

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

perimeter data

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

prapemrosesan data

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

asal data

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

subjek data

Individu yang datanya dikumpulkan dan diproses.

gudang data

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

bahasa definisi database (DDL)

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

bahasa manipulasi basis data (DHTML)

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

DDL

Lihat bahasa definisi database.

ansambel yang dalam

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

pembelajaran mendalam

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

defense-in-depth

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

administrator yang didelegasikan

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

deployment

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

lingkungan pengembangan

Lihat lingkungan.

kontrol detektif

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

pemetaan aliran nilai pengembangan (DVSM)

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

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

kembar digital

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

tabel dimensi

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

musibah

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

pemulihan bencana (DR)

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

DML~

Lihat bahasa manipulasi basis data.

desain berbasis domain

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

DR

Lihat pemulihan bencana.

deteksi drift

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

DVSM

Lihat pemetaan aliran nilai pengembangan.

E

EDA

Lihat analisis data eksplorasi.

EDI

Lihat pertukaran data elektronik.

komputasi tepi

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

pertukaran data elektronik (EDI)

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

enkripsi

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

kunci enkripsi

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

endianness

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

titik akhir

Lihat titik akhir layanan.

layanan endpoint

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

perencanaan sumber daya perusahaan (ERP)

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

enkripsi amplop

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

lingkungan

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

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

epik

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

ERP

Lihat perencanaan sumber daya perusahaan.

analisis data eksplorasi (EDA)

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

F

tabel fakta

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

gagal cepat

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

batas isolasi kesalahan

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

cabang fitur

Lihat cabang.

fitur

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

pentingnya fitur

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

transformasi fitur

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

beberapa tembakan mendorong

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

FGAC

Lihat kontrol akses berbutir halus.

kontrol akses berbutir halus (FGAC)

Penggunaan beberapa kondisi untuk mengizinkan atau menolak permintaan akses.

migrasi flash-cut

Metode migrasi database yang menggunakan replikasi data berkelanjutan melalui pengambilan data perubahan untuk memigrasikan data dalam waktu sesingkat mungkin, alih-alih menggunakan pendekatan bertahap. Tujuannya adalah untuk menjaga downtime seminimal mungkin.

FM

Lihat model pondasi.

model pondasi (FM)

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

G

AI generatif

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

pemblokiran geografis

Lihat pembatasan geografis.

pembatasan geografis (pemblokiran geografis)

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

Alur kerja Gitflow

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

gambar emas

Sebuah snapshot dari sistem atau perangkat lunak yang digunakan sebagai template untuk menyebarkan instance baru dari sistem atau perangkat lunak itu. Misalnya, di bidang manufaktur, gambar emas dapat digunakan untuk menyediakan perangkat lunak pada beberapa perangkat dan membantu meningkatkan kecepatan, skalabilitas, dan produktivitas dalam operasi manufaktur perangkat.

strategi greenfield

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

pagar pembatas

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

H

HA

Lihat ketersediaan tinggi.

migrasi database heterogen

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

ketersediaan tinggi (HA)

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

modernisasi sejarawan

Pendekatan yang digunakan untuk memodernisasi dan meningkatkan sistem teknologi operasional (OT) untuk melayani kebutuhan industri manufaktur dengan lebih baik. Sejarawan

adalah jenis database yang digunakan untuk mengumpulkan dan menyimpan data dari berbagai sumber di pabrik.

data penahanan

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

migrasi database homogen

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

data panas

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

perbaikan terbaru

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

periode hypercare

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

I

IAc

Lihat infrastruktur sebagai kode.

kebijakan berbasis identitas

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

aplikasi idle

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

IIoT

Lihat Internet of Things industri.

infrastruktur yang tidak dapat diubah

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

masuk (masuknya) VPC

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

migrasi inkremental

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

Industri 4.0

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

infrastruktur

Semua sumber daya dan aset yang terkandung dalam lingkungan aplikasi.

infrastruktur sebagai kode (IAc)

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

Internet of Things industri (IIoT)

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

inspeksi VPC

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

Internet of Things (IoT)

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

interpretasi

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

IoT

Lihat Internet of Things.

Perpustakaan informasi TI (ITIL)

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

Manajemen layanan TI (ITSM)

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

ITIL

Lihat perpustakaan informasi TI.

ITSM

Lihat manajemen layanan TI.

L

kontrol akses berbasis label (LBAC)

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

landing zone

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

model bahasa besar (LLM)

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

migrasi besar

Migrasi 300 atau lebih server.

LBAC

Lihat kontrol akses berbasis label.

hak istimewa paling sedikit

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

angkat dan geser

Lihat 7 Rs.

sistem endian kecil

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

LLM

Lihat model bahasa besar.

lingkungan yang lebih rendah

Lihat lingkungan.

M

pembelajaran mesin (ML)

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

cabang utama

Lihat cabang.

malware

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

layanan terkelola

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

sistem eksekusi manufaktur (MES)

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

PETA

Lihat Program Percepatan Migrasi.

mekanisme

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

akun anggota

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

MES

Lihat sistem eksekusi manufaktur.

Transportasi Telemetri Antrian Pesan (MQTT)

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

layanan mikro

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

informasi selengkapnya, lihat Mengintegrasikan layanan mikro dengan menggunakan layanan tanpa AWS server.

arsitektur microservices

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

Program Percepatan Migrasi (MAP)

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

migrasi dalam skala

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

pabrik migrasi

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

metadata migrasi

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

pola migrasi

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

Penilaian Portofolio Migrasi (MPA)

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

Penilaian Kesiapan Migrasi (MRA)

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

strategi migrasi

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

ML

Lihat pembelajaran mesin.

modernisasi

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

penilaian kesiapan modernisasi

Evaluasi yang membantu menentukan kesiapan modernisasi aplikasi organisasi; mengidentifikasi manfaat, risiko, dan dependensi; dan menentukan seberapa baik organisasi dapat mendukung keadaan masa depan aplikasi tersebut. Hasil penilaian adalah cetak biru arsitektur target, peta

jalan yang merinci fase pengembangan dan tonggak untuk proses modernisasi, dan rencana aksi untuk mengatasi kesenjangan yang diidentifikasi. Untuk informasi lebih lanjut, lihat Mengevaluasi kesiapan modernisasi untuk aplikasi di. AWS Cloud

aplikasi monolitik (monolit)

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

MPA

Lihat Penilaian Portofolio Migrasi.

MQTT

Lihat Transportasi Telemetri Antrian Pesan.

klasifikasi multiclass

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

infrastruktur yang bisa berubah

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

O

OAC

Lihat kontrol akses asal.

OAI

Lihat identitas akses asal.

OCM

Lihat manajemen perubahan organisasi.

migrasi offline

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

OI

Lihat integrasi operasi.

OLA

Lihat perjanjian tingkat operasional.

migrasi online

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

OPC-UA

Lihat Komunikasi Proses Terbuka - Arsitektur Terpadu.

Komunikasi Proses Terbuka - Arsitektur Terpadu (OPC-UA)

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

perjanjian tingkat operasional (OLA)

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

Tinjauan Kesiapan Operasional (ORR)

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

teknologi operasional (OT)

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

integrasi operasi (OI)

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

jejak organisasi

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

manajemen perubahan organisasi (OCM)

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

kontrol akses asal (OAC)

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

identitas akses asal (OAI)

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

ORR

Lihat tinjauan kesiapan operasional.

OT

Lihat teknologi operasional.

keluar (jalan keluar) VPC

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

P

batas izin

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

Informasi Identifikasi Pribadi (PII)

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

PII

Lihat informasi yang dapat diidentifikasi secara pribadi.

buku pedoman

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

PLC

Lihat pengontrol logika yang dapat diprogram.

PLM

Lihat manajemen siklus hidup produk.

kebijakan

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

persistensi poliglot

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

penilaian portofolio

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

predikat

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

predikat pushdown

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

kontrol preventif

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

principal

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

privasi berdasarkan desain

Pendekatan rekayasa sistem yang memperhitungkan privasi melalui seluruh proses pengembangan.

zona host pribadi

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

kontrol proaktif

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

manajemen siklus hidup produk (PLM)

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

lingkungan produksi

Lihat lingkungan.

pengontrol logika yang dapat diprogram (PLC)

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

rantai cepat

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

pseudonimisasi

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

publish/subscribe (pub/sub)

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

Q

rencana kueri

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

regresi rencana kueri

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

R

Matriks RACI

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

LAP

Lihat Retrieval Augmented Generation.

ransomware

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

Matriks RASCI

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

RCAC

Lihat kontrol akses baris dan kolom.

replika baca

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

arsitek ulang

Lihat 7 Rs.

tujuan titik pemulihan (RPO)

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

tujuan waktu pemulihan (RTO)

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

refactor

Lihat 7 Rs.

Region

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

regresi

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

rehost

Lihat 7 Rs.

melepaskan

Dalam proses penyebaran, tindakan mempromosikan perubahan pada lingkungan produksi.

memindahkan

Lihat 7 Rs.

memplatform ulang

Lihat 7 Rs.

pembelian kembali

Lihat 7 Rs.

ketahanan

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

kebijakan berbasis sumber daya

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

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

Matriks yang mendefinisikan peran dan tanggung jawab untuk semua pihak yang terlibat dalam kegiatan migrasi dan operasi cloud. Nama matriks berasal dari jenis tanggung jawab yang didefinisikan dalam matriks: bertanggung jawab (R), akuntabel (A), dikonsultasikan (C), dan diinformasikan (I). Jenis dukungan (S) adalah opsional. Jika Anda menyertakan dukungan, matriks disebut matriks RASCI, dan jika Anda mengecualikannya, itu disebut matriks RACI.

kontrol responsif

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

melestarikan

Lihat 7 Rs.

pensiun

Lihat 7 Rs.

Retrieval Augmented Generation (RAG)

Teknologi AI generatif di mana LLM mereferensikan sumber data otoritatif yang berada di luar sumber data pelatihannya sebelum menghasilkan respons. Misalnya, model RAG mungkin

melakukan pencarian semantik dari basis pengetahuan organisasi atau data kustom. Untuk informasi lebih lanjut, lihat Apa itu RAG.

rotasi

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

kontrol akses baris dan kolom (RCAC)

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

RPO

Lihat tujuan titik pemulihan.

RTO

Lihat tujuan waktu pemulihan.

buku runbook

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

D

SAML 2.0

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

PENIPUAN

Lihat kontrol pengawasan dan akuisisi data.

SCP

Lihat kebijakan kontrol layanan.

Rahasia

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

keamanan dengan desain

Pendekatan rekayasa sistem yang memperhitungkan keamanan melalui seluruh proses pengembangan.

kontrol keamanan

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

pengerasan keamanan

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

sistem informasi keamanan dan manajemen acara (SIEM)

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

otomatisasi respons keamanan

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

enkripsi sisi server

Enkripsi data di tujuannya, oleh Layanan AWS yang menerimanya.

kebijakan kontrol layanan (SCP)

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

titik akhir layanan

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

perjanjian tingkat layanan (SLA)

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

indikator tingkat layanan (SLI)

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

tujuan tingkat layanan (SLO)

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

model tanggung jawab bersama

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

SIEM

Lihat informasi keamanan dan sistem manajemen acara.

titik kegagalan tunggal (SPOF)

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

SLA

Lihat perjanjian tingkat layanan.

SLI

Lihat indikator tingkat layanan.

SLO

Lihat tujuan tingkat layanan.

split-and-seed model

Pola untuk menskalakan dan mempercepat proyek modernisasi. Ketika fitur baru dan rilis produk didefinisikan, tim inti berpisah untuk membuat tim produk baru. Ini membantu meningkatkan kemampuan dan layanan organisasi Anda, meningkatkan produktivitas pengembang, dan mendukung inovasi yang cepat. Untuk informasi lebih lanjut, lihat Pendekatan bertahap untuk memodernisasi aplikasi di. AWS Cloud

SPOF

Lihat satu titik kegagalan.

skema bintang

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

pola ara pencekik

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

subnet

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

kontrol pengawasan dan akuisisi data (SCADA)

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

enkripsi simetris

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

pengujian sintetis

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

sistem prompt

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

T

tag

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

variabel target

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

daftar tugas

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

lingkungan uji

Lihat lingkungan.

pelatihan

Untuk menyediakan data bagi model ML Anda untuk dipelajari. Data pelatihan harus berisi jawaban yang benar. Algoritma pembelajaran menemukan pola dalam data pelatihan yang

memetakan atribut data input ke target (jawaban yang ingin Anda prediksi). Ini menghasilkan model ML yang menangkap pola-pola ini. Anda kemudian dapat menggunakan model ML untuk membuat prediksi pada data baru yang Anda tidak tahu targetnya.

gerbang transit

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

alur kerja berbasis batang

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

akses tepercaya

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

penyetelan

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

tim dua pizza

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

U

waswas

Sebuah konsep yang mengacu pada informasi yang tidak tepat, tidak lengkap, atau tidak diketahui yang dapat merusak keandalan model ML prediktif. Ada dua jenis ketidakpastian:

ketidakpastian epistemik disebabkan oleh data yang terbatas dan tidak lengkap, sedangkan ketidakpastian aleatorik disebabkan oleh kebisingan dan keacakan yang melekat dalam data.

tugas yang tidak terdiferensiasi

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

lingkungan atas

Lihat lingkungan.

V

menyedot debu

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

kendali versi

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

Peering VPC

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

kerentanan

Kelemahan perangkat lunak atau perangkat keras yang membahayakan keamanan sistem.

W

cache hangat

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

data hangat

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

fungsi jendela

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

beban kerja

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

aliran kerja

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

CACING

Lihat menulis sekali, baca banyak.

WQF

Lihat AWS Kerangka Kualifikasi Beban Kerja.

tulis sekali, baca banyak (WORM)

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

Z

eksploitasi zero-day

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

kerentanan zero-day

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

bisikan zero-shot

Memberikan LLM dengan instruksi untuk melakukan tugas tetapi tidak ada contoh (tembakan) yang dapat membantu membimbingnya. LLM harus menggunakan pengetahuan pra-terlatih untuk menangani tugas. Efektivitas bidikan nol tergantung pada kompleksitas tugas dan kualitas prompt. Lihat juga beberapa bidikan yang diminta.

aplikasi zombie

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

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