Tes CI/CD lakmus: Apakah pipa Anda sepenuhnya CI/CD?

AWS Bimbingan Preskriptif

AWS Bimbingan Preskriptif: Tes CI/CD lakmus: Apakah pipa Anda sepenuhnya CI/CD?

Table of Contents

Tes CI/CD lakmus: Apakah pipa Anda sepenuhnya CI/CD?

Steven Guggenheimer dan Ananya Koduri, Amazon Web Services ()AWS

Agustus 2023 (sejarah dokumen)

Apakah pipa Anda otomatis? Ini pertanyaan sederhana, tetapi banyak organisasi mendekati jawabannya terlalu sederhana. Jawabannya jauh lebih rumit daripada ya atau tidak.

Inovasi dalam teknologi terjadi terus-menerus, dan terkadang sulit bagi organisasi untuk mengikutinya. Apakah hal baru ini iseng-iseng, atau apakah ini hal besar berikutnya? Haruskah saya merombak praktik saya saat ini, atau haruskah saya menunggu? Seringkali, pada saat menjadi jelas bahwa sesuatu memang hal besar berikutnya, Anda dapat menemukan diri Anda bermain catchup. Integrasi berkelanjutan dan pengiriman berkelanjutan (CI/CD) akan tetap ada, tetapi tidak selalu seperti itu. Banyak orang membutuhkan waktu lama untuk diyakinkan, dan beberapa orang masih perlu lebih meyakinkan.

CI/CD is the process of automating the source, build, test, staging, and production stages of the software release process, and it is commonly described as a pipeline. Today, the cost savings and speed of CI/CDotomatisasi telah meyakinkan sebagian besar organisasi tentang nilainya. Tetapi beralih ke pendekatan baru ini bukanlah tugas yang mudah. Anda perlu memastikan staf Anda memiliki pelatihan yang tepat, Anda perlu meningkatkan beberapa sumber daya, dan kemudian Anda perlu menguji, menguji, menguji. Ada banyak yang harus dilakukan. Dalam kebanyakan kasus, Anda ingin membuat perubahan ini secara bertahap untuk membantu organisasi Anda beradaptasi.

Tujuan dari dokumen ini adalah untuk mendefinisikan apa artinya memiliki CI/CD proses yang sepenuhnya. Ini menyediakan alat untuk mengevaluasi proses Anda sendiri dan menyajikan jalan ke depan untuk proses yang belum ada. Jalan ke depan ini jarang merupakan pertobatan dalam semalam. Proses-proses ini kompleks dan bergantung pada banyak faktor, termasuk keahlian karyawan saat ini dan tuntutan infrastruktur saat ini. Kami menyarankan Anda memprioritaskan dan membuat perubahan kecil dan bertahap.

Tujuan

Berikut ini adalah manfaat potensial dari penerapan rekomendasi dalam panduan ini:

- Efisiensi — Proses CI/CD penyebaran sepenuhnya dapat mengurangi kompleksitas, beban kerja, dan jam yang tak terhitung jumlahnya menghabiskan debugging, melakukan proses manual, dan

memelihara. Untuk informasi lebih lanjut, lihat Manfaat pengiriman berkelanjutan. Menurut sebuah posting TechAhead blog, implementasi CI/CD proses dapat menghasilkan penghematan sekitar 20% dalam waktu, tenaga, dan sumber daya.

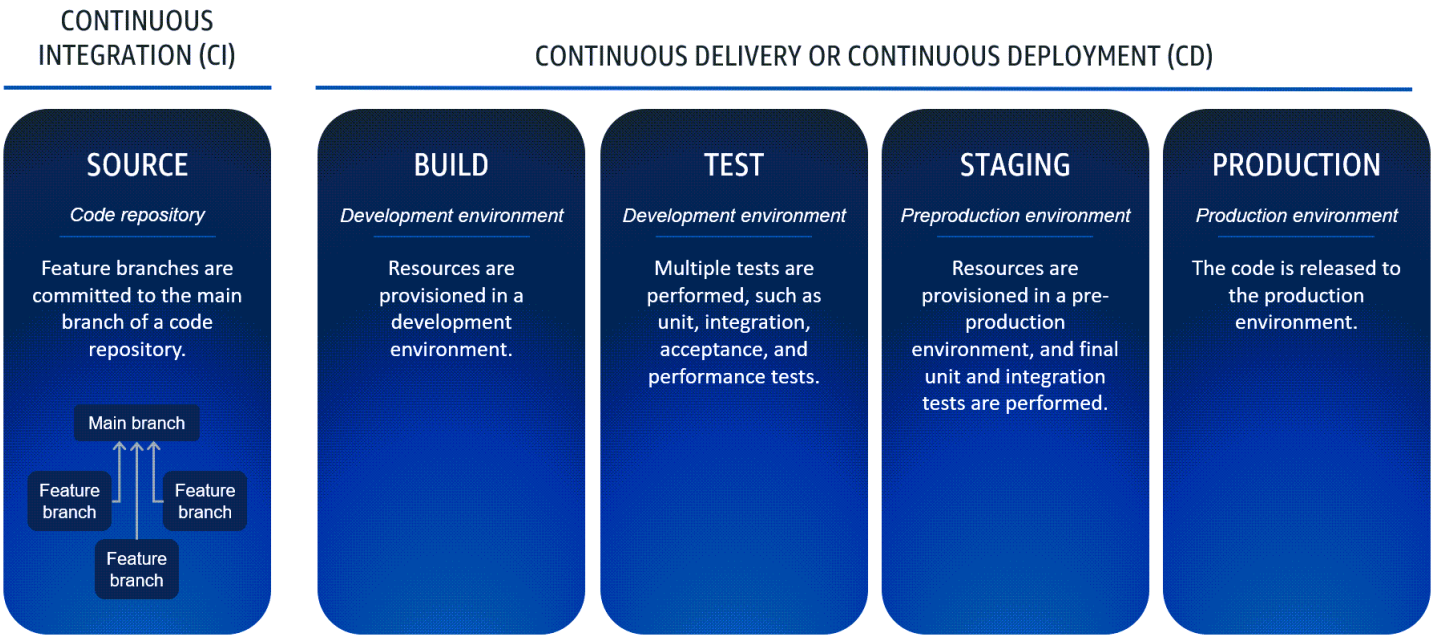
- Pengurangan biaya — Menurut laporan Forbes Insight, “Tiga dari empat eksekutif setuju bahwa jumlah waktu, uang, dan sumber daya yang dihabiskan untuk pemeliharaan dan manajemen yang berkelanjutan — versus pengembangan proyek baru atau inisiatif baru — memengaruhi daya saing keseluruhan organisasi mereka.” Semakin pendek siklus pengembangan, semakin tinggi kemungkinan organisasi Anda dapat memenuhi time-to-market tujuan ambisius dan meraih peluang yang tepat pada waktu yang tepat.
- Kecepatan — Biasanya, CI/CD pipeline is able to release software changes to customers within a few hours. Especially in cases with quick fault isolations and small patch pushes, the CI/CD pipa sepenuhnya membantu meningkatkan mean time to recovery (MTTR). Untuk informasi selengkapnya, lihat Mengurangi MTTR.
- Keamanan — CI/CD Saluran pipa sepenuhnya juga mengamankan proses pelepasan dengan mengurangi kemungkinan titik masuk untuk serangan dan mengurangi risiko kesalahan manusia. Keuntungan keamanan yang datang dengan CI/CD jaringan pipa otomatis sepenuhnya membantu menghindari konsekuensi mahal dari pelanggaran data, pemadaman layanan, dan banyak lagi.
- Mengurangi gesekan - Pengembang lebih puas ketika mereka dapat menghabiskan lebih banyak waktu untuk membuat fitur hebat dan lebih sedikit waktu yang terperosok dalam siklus pemeliharaan dan debugging tanpa akhir. Untuk organisasi, ini berarti memperoleh dan mempertahankan talenta terbaik untuk jangka waktu yang lebih lama.
- Kode kualitas unggul — Pengembang merilis kode ke dalam repositori bersama dalam batch kecil, yang memungkinkan mereka untuk melakukan pengujian paralel (BrowserStack posting blog). Alih-alih bekerja secara terpisah, mereka sering berbagi build dengan tim, dan mereka berkolaborasi untuk mengidentifikasi bug kritis. Ini memberikan dukungan bagi pengembang, yang membantu mencegah kode buruk membuatnya menjadi produksi. Support dari rekan pengembang berkontribusi pada rilis berkualitas tinggi dan mendorong pertumbuhan organisasi.
- Pemeliharaan - Pemeliharaan dan pembaruan adalah bagian penting untuk membuat produk yang hebat. Namun, jangan turunkan sistem selama waktu lalu lintas puncak. Anda dapat menggunakan CI/CD pipeline untuk melakukan pemeliharaan selama jam penggunaan rendah yang meminimalkan waktu henti dan dampak kinerja.

Memahami CI/CD

Integrasi berkelanjutan dan pengiriman berkelanjutan (CI/CD) adalah proses mengotomatisasi siklus hidup rilis perangkat lunak. Dalam beberapa kasus, D in juga CI/CD dapat berarti penyebaran. Perbedaan antara pengiriman berkelanjutan dan penerapan berkelanjutan terjadi ketika Anda merilis perubahan pada lingkungan produksi. Dengan pengiriman berkelanjutan, persetujuan manual diperlukan sebelum mempromosikan perubahan produksi. Penerapan berkelanjutan menampilkan aliran tanpa gangguan melalui keseluruhan pipeline, dan tidak ada persetujuan eksplisit yang diperlukan. Karena strategi ini membahas CI/CD konsep umum, rekomendasi dan informasi yang diberikan berlaku untuk pengiriman berkelanjutan dan pendekatan penyebaran berkelanjutan.

CI/CD automates much or all of the manual processes traditionally required to get new code from a commit into production. A CI/CD pipeline encompasses the source, build, test, staging, and production stages. In each stage, the CI/CD pipelines provisions any infrastructure that is needed to deploy or test the code. By using a CI/CDpipeline, tim pengembangan dapat membuat perubahan pada kode yang kemudian diuji secara otomatis dan didorong ke penerapan.

Mari kita tinjau CI/CD proses dasar sebelum membahas beberapa cara yang dapat Anda, secara sadar atau tidak sadar, menyimpang dari tahap dan aktivitas penuh CI/CD. The following diagram shows the CI/CD di setiap tahap.



Tentang integrasi berkelanjutan

Integrasi berkelanjutan terjadi dalam repositori kode, seperti repositori Git di. GitHub Anda memperlakukan satu cabang utama sebagai sumber kebenaran untuk basis kode, dan Anda membuat cabang berumur pendek untuk pengembangan fitur. Anda mengintegrasikan cabang fitur ke cabang utama saat Anda siap untuk menerapkan fitur ke lingkungan atas. Cabang fitur tidak pernah digunakan langsung ke lingkungan atas. Untuk informasi selengkapnya, lihat Pendekatan berbasis batang dalam panduan ini.

Proses integrasi berkelanjutan

1. Pengembang membuat cabang baru dari cabang utama.
2. Pengembang membuat perubahan dan membangun dan menguji secara lokal.
3. Ketika perubahan sudah siap, pengembang membuat permintaan tarik (GitHub dokumentasi) dengan cabang utama sebagai tujuan.
4. Kode ditinjau.
5. Ketika kode disetujui, itu digabungkan ke cabang utama.

Tentang pengiriman berkelanjutan

Pengiriman berkelanjutan terjadi di lingkungan yang terisolasi, seperti lingkungan pengembangan dan lingkungan produksi. Tindakan yang terjadi di setiap lingkungan dapat bervariasi. Seringkali, salah satu tahap pertama digunakan untuk membuat pembaruan pada pipa itu sendiri sebelum melanjutkan. Hasil akhir dari penerapan adalah bahwa setiap lingkungan diperbarui dengan perubahan terbaru. Jumlah lingkungan pengembangan untuk bangunan dan pengujian juga bervariasi, tetapi kami sarankan Anda menggunakan setidaknya dua. Dalam pipa, setiap lingkungan diperbarui dalam urutan signifikansinya, berakhir dengan lingkungan yang paling penting, lingkungan produksi.

Proses pengiriman berkelanjutan

Bagian pengiriman berkelanjutan dari pipeline dimulai dengan menarik kode dari cabang utama repositori sumber dan meneruskannya ke tahap pembuatan. Dokumen infrastruktur sebagai kode (IAc) untuk repositori menguraikan tugas-tugas yang dilakukan di setiap tahap. Meskipun menggunakan dokumen IAc tidak wajib, layanan atau alat IAC, seperti AWS CloudFormationatau AWS Cloud Development Kit (AWS CDK), sangat disarankan. Langkah-langkah yang paling umum meliputi:

1. Tes unit
2. Membangun kode
3. Penyediaan sumber daya
4. Tes integrasi

Jika terjadi kesalahan atau pengujian apa pun gagal pada tahap apa pun dalam pipa, tahap saat ini kembali ke keadaan sebelumnya, dan pipa dihentikan. Perubahan selanjutnya harus dimulai di repositori kode dan melalui proses penuh CI/CD .

Tes untuk CI/CD jaringan pipa

Dua jenis pengujian otomatis yang biasa disebut dalam pipeline penyebaran adalah pengujian unit dan tes integrasi. Namun, ada banyak jenis pengujian yang dapat Anda jalankan pada basis kode dan lingkungan pengembangan. Arsitektur Referensi Pipeline AWS Deployment mendefinisikan jenis pengujian berikut:

- Unit test — Tes ini membangun dan menjalankan kode aplikasi untuk memverifikasi bahwa itu bekerja sesuai dengan harapan. Mereka mensimulasikan semua dependensi eksternal yang digunakan dalam basis kode. Contoh alat uji unit termasuk JUnit, Jest, dan pytest.
- Tes integrasi — Pengujian ini memverifikasi bahwa aplikasi memenuhi persyaratan teknis dengan menguji lingkungan pengujian yang disediakan. Contoh alat uji integrasi termasuk Mentimun, VRest NG, dan tes integ (untuk). AWS CDK
- Tes penerimaan - Pengujian ini memverifikasi bahwa aplikasi memenuhi persyaratan pengguna dengan menguji lingkungan pengujian yang disediakan. Contoh alat uji penerimaan termasuk Cypress dan Selenium.
- Tes sintetis — Tes ini berjalan terus menerus di latar belakang untuk menghasilkan lalu lintas dan memverifikasi bahwa sistemnya sehat. Contoh alat uji sintetis termasuk Amazon CloudWatch Synthetics dan Dynatrace Synthetic Monitoring.
- Uji kinerja - Tes ini mensimulasikan kapasitas produksi. Mereka menentukan apakah aplikasi memenuhi persyaratan kinerja dan membandingkan metrik dengan kinerja masa lalu. Contoh alat uji kinerja termasuk Apache JMeter, Locust, dan Gatling.
- Tes ketahanan — Juga dikenal sebagai pengujian kekacauan, tes ini menyuntikkan kegagalan ke lingkungan untuk mengidentifikasi area risiko. Periode ketika kegagalan disuntikkan kemudian dibandingkan dengan periode tanpa kegagalan. Contoh alat uji ketahanan termasuk AWS Fault Injection Servicedan Gremlin.

- Tes keamanan aplikasi statis (SAST) — Tes ini menganalisis kode untuk pelanggaran keamanan, seperti injeksi SQL atau cross-site scripting (XSS). Contoh alat SAST termasuk Amazon CodeGuru, SonarQube, dan Checkmarx.
- Dynamic Application Security Test (DAST) — Tes ini juga dikenal sebagai pengujian penetrasi atau pengujian pena. Mereka mengidentifikasi kerentanan, seperti injeksi SQL atau XSS di lingkungan pengujian yang disediakan. Contoh alat DAST termasuk Zed Attack Proxy (ZAP) dan HCL. AppScan Untuk informasi selengkapnya, lihat Pengujian Penetrasi.

Tidak semua CI/CD jaringan pipa sepenuhnya menjalankan semua tes ini. Namun, minimal, pipeline harus menjalankan pengujian unit dan pengujian SAST pada basis kode serta tes integrasi dan penerimaan pada lingkungan pengujian.

Metrik untuk jaringan pipa CI/CD

Menurut Arsitektur Referensi Pipeline AWS Deployment, Anda harus, setidaknya, melacak empat metrik berikut untuk CI/CD pipeline:

- Lead time — Jumlah rata-rata waktu yang dibutuhkan untuk satu komitmen untuk masuk ke dalam produksi. Kami merekomendasikan untuk menargetkan lead time antara 1 jam dan 1 hari, yang sesuai untuk kasus penggunaan Anda.
- Frekuensi penyebaran — Jumlah penyebaran produksi dalam periode waktu tertentu. Kami merekomendasikan penargetan frekuensi penerapan antara beberapa kali setiap hari hingga dua kali setiap minggu, yang sesuai untuk kasus penggunaan Anda.
- Mean time between failure (MTBF) — Rata-rata jumlah waktu antara awal pipa yang berhasil dan dimulainya pipa yang gagal. Kami merekomendasikan untuk menargetkan MTBF setinggi mungkin. Untuk informasi selengkapnya, lihat Meningkatkan MTBF.
- Mean time to recovery (MTTR) — Jumlah rata-rata waktu antara awal pipa yang gagal dan dimulainya pipa sukses berikutnya. Kami merekomendasikan penargetan MTTR yang serendah mungkin. Untuk informasi selengkapnya, lihat Mengurangi MTTR.

Metrik ini membantu tim melacak kemajuan mereka untuk menjadi CI/CD sepenuhnya. Tim harus melakukan diskusi terbuka dengan pemangku kepentingan organisasi mengenai apa tujuan optimal yang seharusnya. Situasi dan kebutuhan sangat bervariasi dari organisasi ke organisasi, dan bahkan dari tim ke tim.

Sangat penting untuk diingat bahwa perubahan yang cepat dan drastis biasanya meningkatkan risiko masalah yang timbul. Tetapkan tujuan untuk mencapai peningkatan kecil dan bertahap. Waktu tunggu optimal yang umum untuk CI/CD jaringan pipa sepenuhnya kurang dari 3 jam. Sebuah tim yang memulai dengan lead time 5,2 hari harus menargetkan pengurangan satu hari setiap beberapa minggu. Setelah tim ini mencapai lead time satu hari atau kurang, mereka dapat tinggal di sana selama beberapa bulan dan pindah ke lead time yang lebih agresif hanya jika tim dan pemangku kepentingan organisasi menganggapnya perlu.

Bagaimana CI/CD proses sepenuhnya berbeda

Pipa CI/CD menggunakan alur kerja berbasis batang modern, di mana pengembang menggabungkan pembaruan kecil dan sering ke cabang utama (atau trunk) yang dibangun dan diuji melalui bagian CD dari pipa. CI/CD Alur kerja ini telah menggantikan alur kerja Gitflow, di mana cabang pengembangan dan rilis dipisahkan oleh jadwal rilis. Di banyak organisasi, Gitflow masih merupakan metode kontrol dan penyebaran versi yang populer. Namun, sekarang dianggap warisan, dan dapat menjadi tantangan untuk diintegrasikan ke dalam CI/CD pipa.

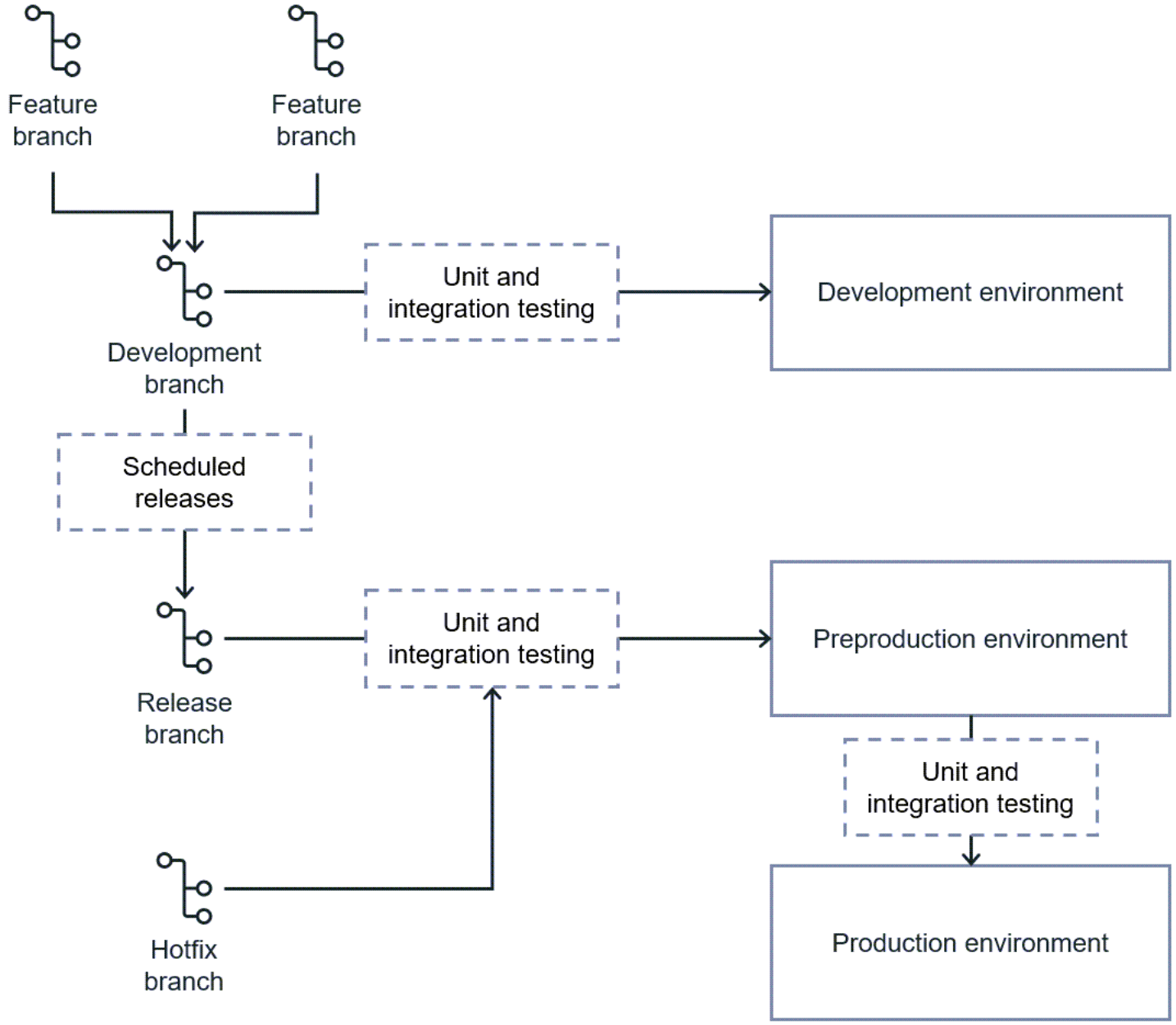
Bagi banyak organisasi, transisi dari alur kerja Gitflow ke alur kerja berbasis batang tidak lengkap, dan hasilnya adalah mereka terjebak di suatu tempat di sepanjang jalan dan tidak pernah sepenuhnya bermigrasi ke CI/CD. Entah bagaimana, jaringan pipa mereka akhirnya menempel pada sisa-sisa tertentu dari alur kerja warisan, terjebak dalam keadaan transisi antara masa lalu dan sekarang. Tinjau perbedaan dalam alur kerja Git, lalu pelajari cara menggunakan alur kerja lama dapat memengaruhi hal-hal berikut:

- Integritas lingkungan
- Rilis
- Keamanan

Untuk mempermudah mengidentifikasi sisa-sisa alur kerja Git lama dalam konfigurasi modern, mari kita bandingkan Gitflow dengan pendekatan modern berbasis batang.

Pendekatan Gitflow

Gambar berikut menunjukkan alur kerja Gitflow. Pendekatan Gitflow menggunakan beberapa cabang untuk melacak beberapa versi kode yang berbeda secara bersamaan. Anda menjadwalkan rilis pembaruan ke aplikasi untuk beberapa titik di masa mendatang sementara pengembang masih mengerjakan versi kode saat ini. Repositori berbasis Trunk dapat menggunakan flag fitur untuk mencapai ini, tetapi itu dibangun ke Gitflow secara default.

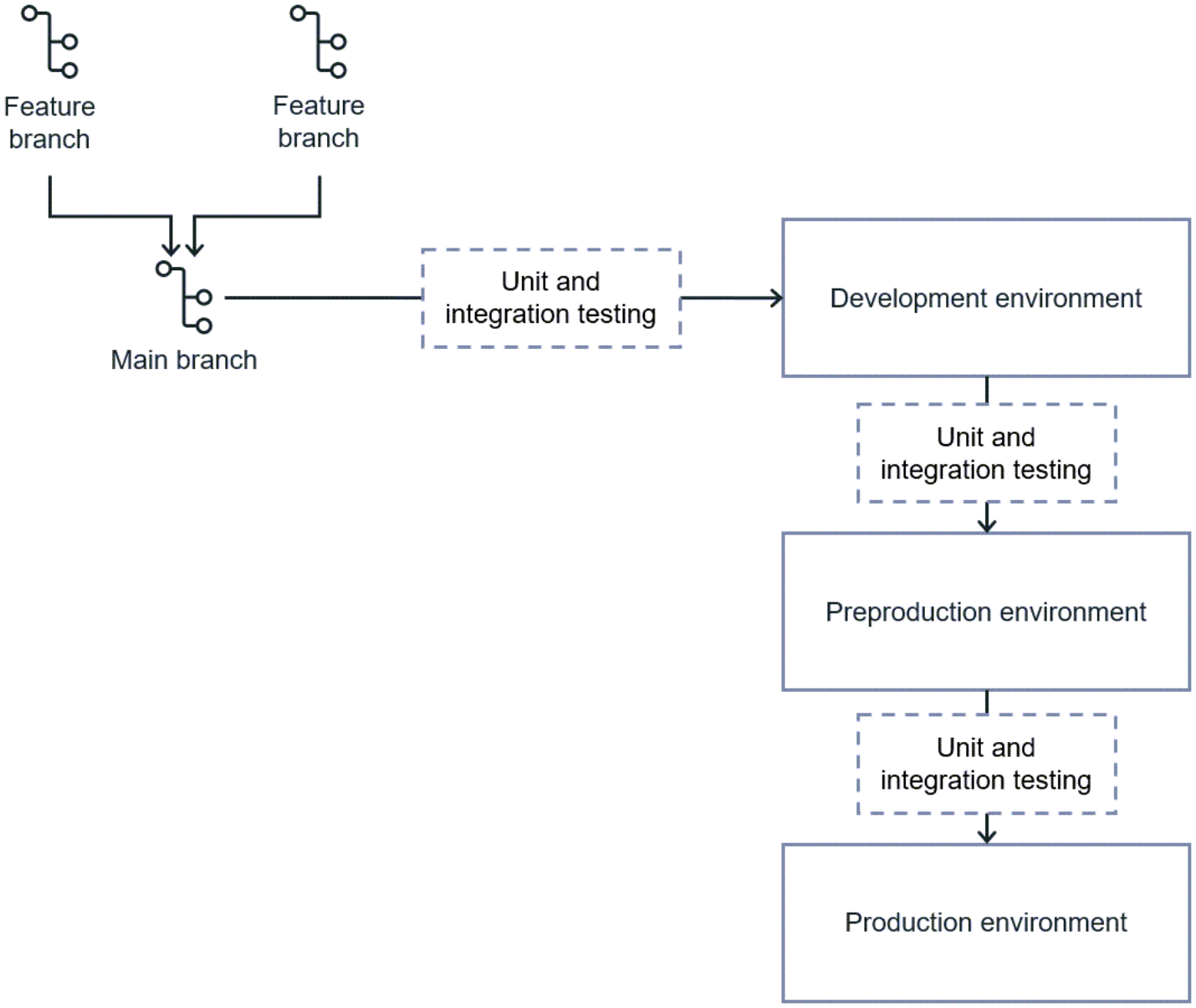


Salah satu hasil dari pendekatan Gitflow adalah bahwa lingkungan aplikasi biasanya tidak sinkron. Dalam implementasi Gitflow standar, lingkungan pengembangan mencerminkan status kode saat ini sementara lingkungan praproduksi dan produksi tetap beku pada status basis kode dari rilis terbaru.

Ini memperumit hal-hal ketika cacat muncul di lingkungan produksi karena basis kode tempat pengembang bekerja tidak dapat digabungkan ke dalam produksi tanpa mengekspos fitur yang belum dirilis. Cara Gitflow menangani situasi ini adalah dengan menggunakan perbaikan terbaru. Cabang hotfix dibuat dari cabang rilis dan kemudian diterapkan langsung ke lingkungan atas. Cabang hotfix kemudian digabungkan ke dalam cabang pengembangan untuk menjaga kode tetap terkini.

Pendekatan berbasis batang

Gambar berikut menunjukkan alur kerja berbasis batang. Dalam alur kerja berbasis batang, 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. Tes unit dan integrasi terjadi antara setiap lingkungan.



Menggunakan alur kerja ini, semua lingkungan mengoperasikan basis kode yang sama. Tidak perlu cabang perbaikan terbaru untuk lingkungan atas karena Anda dapat menerapkan perubahan di cabang utama tanpa mengekspos fitur yang belum dirilis. Cabang utama selalu dianggap stabil,

bebas dari cacat, dan siap dilepaskan. Ini membantu Anda mengintegrasikannya sebagai sumber untuk CI/CD pipeline, yang dapat secara otomatis menguji dan menerapkan basis kode Anda melalui semua lingkungan di pipeline Anda.

Manfaat integritas lingkungan dari pendekatan berbasis batang

Seperti yang diketahui banyak pengembang, satu perubahan kode terkadang dapat menciptakan efek kupu-kupu (artikel American Scientist), di mana penyimpangan kecil yang tampaknya tidak terkait memicu reaksi berantai yang menyebabkan hasil yang tidak terduga. Pengembang kemudian harus menyelidiki sepenuhnya untuk menemukan akar masalahnya.

Ketika para ilmuwan melakukan percobaan, mereka memisahkan subjek uji menjadi dua kelompok: kelompok eksperimen dan kelompok kontrol. Tujuannya adalah untuk membuat kelompok eksperimen dan kelompok kontrol benar-benar identik kecuali untuk hal yang diuji dalam percobaan. Ketika sesuatu terjadi pada kelompok eksperimen yang tidak terjadi pada kelompok kontrol, satu-satunya penyebab adalah hal yang sedang diuji.

Pikirkan perubahan dalam penyebaran sebagai kelompok eksperimen, dan pikirkan setiap lingkungan sebagai kelompok kontrol yang terpisah. Hasil pengujian di lingkungan yang lebih rendah hanya dapat diandalkan ketika kontrolnya sama dengan di lingkungan atas. Semakin banyak lingkungan menyimpang, semakin besar peluang untuk menemukan cacat di lingkungan atas. Dengan kata lain, jika perubahan kode akan gagal dalam produksi, kami lebih suka mereka gagal dalam versi beta terlebih dahulu sehingga tidak pernah sampai ke produksi. Inilah sebabnya mengapa setiap upaya harus dilakukan untuk menjaga setiap lingkungan, dari lingkungan pengujian terendah hingga produksi itu sendiri, sinkron. Ini disebut integritas lingkungan.

Tujuan dari setiap CI/CD proses sepenuhnya adalah untuk menemukan masalah sedini mungkin. Melestarikan integritas lingkungan dengan menggunakan pendekatan berbasis batang hampir dapat menghilangkan kebutuhan akan perbaikan terbaru. Dalam alur kerja berbasis batang, jarang masalah muncul pertama kali di lingkungan produksi.

Dalam pendekatan Gitflow, setelah hotfix diterapkan langsung ke lingkungan atas, kemudian ditambahkan ke cabang pengembangan. Ini mempertahankan perbaikan untuk rilis future. Namun, perbaikan terbaru dikembangkan dan diuji langsung dari keadaan aplikasi saat ini. Bahkan jika perbaikan terbaru bekerja dengan sempurna dalam produksi, ada kemungkinan masalah akan muncul ketika berinteraksi dengan fitur yang lebih baru di cabang pengembangan. Karena menerapkan perbaikan terbaru untuk perbaikan terbaru biasanya tidak diinginkan, ini menyebabkan pengembang menghabiskan waktu ekstra untuk mencoba memperbaiki perbaikan terbaru ke

lingkungan pengembangan. Dalam banyak kasus, ini dapat menyebabkan utang teknis yang signifikan dan mengurangi stabilitas lingkungan pembangunan secara keseluruhan.

Ketika kegagalan terjadi di suatu lingkungan, semua perubahan diputar kembali sehingga lingkungan dikembalikan ke keadaan sebelumnya. Setiap perubahan pada basis kode harus memulai pipeline lagi dari tahap pertama. Ketika masalah muncul di lingkungan produksi, perbaikan harus melalui seluruh pipa juga. Waktu ekstra yang diperlukan untuk melewati lingkungan yang lebih rendah biasanya dapat diabaikan dibandingkan dengan masalah yang dihindari dengan menggunakan pendekatan ini. Karena seluruh tujuan lingkungan yang lebih rendah adalah untuk menangkap kesalahan sebelum mencapai produksi, melewati lingkungan ini melalui pendekatan Gitflow adalah risiko yang tidak efisien dan tidak perlu.

Rilis manfaat dari pendekatan berbasis batang

Salah satu hal yang sering membuat perbaikan terbaru diperlukan adalah bahwa, dalam alur kerja lama, status aplikasi yang sedang dikerjakan pengembang mungkin berisi beberapa fitur yang belum dirilis yang belum aktif dalam produksi. Lingkungan produksi dan lingkungan pengembangan hanya menjadi sinkron ketika rilis terjadwal terjadi, dan kemudian mereka segera mulai menyimpang lagi hingga rilis terjadwal berikutnya.

Memiliki rilis terjadwal dimungkinkan dalam CI/CD proses penuh. Anda dapat menunda rilis kode ke produksi dengan menggunakan flag fitur. Namun, CI/CD proses yang sepenuhnya memungkinkan lebih banyak fleksibilitas dengan membuat rilis terjadwal tidak perlu. Bagaimanapun, kontinu adalah kata kunci dalam CI/CD, dan itu menunjukkan bahwa perubahan dilepaskan saat mereka siap. Hindari mempertahankan lingkungan rilis terpisah yang hampir selalu tidak sinkron dengan lingkungan pengujian yang lebih rendah.

Jika pipa tidak sepenuhnya CI/CD, divergensi antara lingkungan atas dan bawah biasanya terjadi pada tingkat cabang. Pengembang bekerja di cabang pengembangan dan memelihara cabang rilis terpisah yang diperbarui hanya ketika tiba waktunya untuk rilis terjadwal. Saat cabang pelepasan dan cabang pengembangan berbeda, komplikasi lain dapat muncul.

Selain lingkungan yang tidak sinkron, karena pengembang bekerja di cabang pengembangan dan menjadi terbiasa dengan status aplikasi yang jauh di depan apa yang ada dalam produksi, mereka harus menyesuaikan kembali dengan keadaan produksi setiap kali masalah muncul di sana. Keadaan cabang pengembangan bisa menjadi banyak fitur di depan produksi. Ketika pengembang bekerja di cabang itu setiap hari, sulit untuk mengingat apa yang dirilis dan tidak dirilis ke produksi. Ini menambah risiko bahwa bug baru akan diperkenalkan saat dalam proses memperbaiki bug lainnya.

Hasil ini adalah siklus perbaikan yang tampaknya tak ada habisnya yang memperpanjang jadwal dan menunda rilis fitur selama berminggu-minggu, berbulan-bulan, atau bahkan bertahun-tahun.

Manfaat keamanan dari pendekatan berbasis batang

CI/CD Proses penuh menyediakan satu sumber pendekatan kebenaran yang sepenuhnya otomatis untuk penyebaran. Pipa memiliki satu titik masuk. Pembaruan perangkat lunak memasuki pipeline di awal dan diteruskan apa adanya dari satu lingkungan ke lingkungan berikutnya. Jika masalah ditemukan pada tahap apa pun dalam pipeline, perubahan kode yang memperbaikinya harus melalui proses yang sama dan dimulai pada tahap pertama. Mengurangi titik masuk dalam pipa juga mengurangi kemungkinan cara kerentanan dapat dimasukkan ke dalam pipa.

Selain itu, karena titik masuk adalah titik terjauh dari lingkungan produksi, ini secara drastis mengurangi kemungkinan kerentanan mencapai produksi. Jika Anda menerapkan proses persetujuan manual dalam pipeline CI/CD sepenuhnya, Anda masih dapat mengizinkan pengambilan keputusan go atau no-go tentang apakah perubahan dipromosikan ke lingkungan berikutnya. Pengambil keputusan belum tentu orang yang sama yang menyebarkan perubahan. Ini memisahkan tanggung jawab untuk penyebaran perubahan kode dan pemberi persetujuan perubahan tersebut. Ini juga membuatnya lebih layak bagi pemimpin organisasi yang kurang teknis untuk melakukan peran pemberi persetujuan.

Terakhir, satu titik entri membantu Anda membatasi akses tulis ke konsol antarmuka pengguna (UI) lingkungan produksi untuk beberapa atau bahkan nol pengguna. Dengan mengurangi jumlah pengguna yang dapat membuat perubahan manual di konsol, Anda mengurangi risiko peristiwa keamanan. Kemampuan untuk mengelola konsol secara manual di lingkungan produksi jauh lebih diperlukan dalam alur kerja lama daripada dalam pendekatan CI/CD otomatis. Perubahan manual ini lebih sulit dilacak, ditinjau, dan diuji. Mereka biasanya dilakukan untuk menghemat waktu, tetapi dalam jangka panjang, mereka menambahkan utang teknis yang signifikan ke proyek.

Masalah keamanan konsol tidak selalu disebabkan oleh aktor jahat. Banyak masalah yang terjadi di konsol tidak disengaja. Paparan keamanan yang tidak disengaja sangat umum, dan telah menyebabkan munculnya model keamanan zero-trust. Model ini menyatakan, sebagian, bahwa kecelakaan keamanan lebih kecil kemungkinannya ketika bahkan staf internal memiliki akses sesedikit mungkin, juga dikenal sebagai izin hak istimewa terkecil. Melestarikan integritas lingkungan produksi dengan membatasi semua proses ke pipa otomatis secara praktis menghilangkan risiko masalah keamanan terkait konsol.

Uji lakmus untuk jaringan pipa CI/CD

Dalam kimia, lakmus paper adalah strip tipis kertas yang diolah dengan pewarna merah atau biru khusus yang digunakan untuk menentukan keasaman suatu zat. Asam mengubah kertas lakmus biru menjadi merah, basa berubah menjadi merah kertas lakmus biru, dan zat netral tidak mempengaruhi warna kertas sama sekali.

Cara kertas lakmus menentukan keasaman adalah dengan mengukur tingkat pH suatu zat. Jika tingkat pH lebih tinggi dari 8, itu asam; jika di bawah 5, itu basa; dan jika antara 5 dan 8, itu netral. Demikian pula, tes CI/CD lakmus membantu Anda mengukur CI/CD tingkat pipa Anda.

Untuk menguji apakah pipeline Anda sepenuhnya CI/CD

1. Mulailah dengan skor 0.
2. Jawab setiap pertanyaan berikut, dan tambahkan 1 ke skor Anda untuk setiap kali Anda menjawab ya:

    - Apakah repositori kita masing-masing memiliki tepat satu cabang utama yang digunakan untuk menyebarkan ke lingkungan?
    - Apakah kita sering mengkomit kode ke cabang utama dan menghindari cabang fitur yang berjalan lama?
    - Apakah pipa kami memiliki satu titik masuk? Dengan kata lain, apakah pipeline kami menarik kode dari setiap repositori tepat satu kali?
    - Apakah kita memiliki lebih dari satu lingkungan penerapan?
    - Ketika pipeline tidak berjalan, apakah lingkungan atas dan bawah kita umumnya sinkron?
    - Apakah kita menjalankan tes pada kode sebelum menerapkan?
    - Apakah kita menjalankan pengujian pada lingkungan sebelum mempromosikan ke lingkungan berikutnya?
    - Apakah pipa kami melakukan rollback penuh dan keluar setelah kegagalan?
    - Apakah pipeline kami dimulai ulang dari langkah pertama saat pulih dari kegagalan?
    - Apakah kami mengikuti proses yang sama untuk memperbaiki bug dalam produksi yang kami lakukan untuk merilis fitur ke produksi?
    - Apakah kita menggunakan beberapa bentuk templat infrastruktur sebagai kode (IAc) untuk menyebarkan kode?

3. Jawab setiap pertanyaan berikut, dan tambahkan 1 ke skor Anda untuk setiap kali Anda menjawab tidak:
   - Apakah kita pernah menyebarkan langsung ke lingkungan penerapan dari cabang selain cabang utama?
   - Apakah kita pernah menyebarkan langsung dari cabang mana pun ke lingkungan atas atau produksi?
   - Apakah kita sering menemukan bug di lingkungan atas yang tidak ada di lingkungan yang lebih rendah?
   - Apakah kita pernah melewati lingkungan yang lebih rendah selama penerapan?
   - Apakah kita menunggu hingga waktu rilis yang dijadwalkan untuk diterapkan ke produksi?
   - Apakah kami secara teratur melakukan pembaruan di konsol lingkungan produksi?
   - Apakah ada langkah penerapan manual yang harus dilakukan di konsol lingkungan produksi untuk menyelesaikan penerapan?
   - Apakah lebih dari satu orang memiliki akses tulis ke lingkungan produksi?
   - Apakah lebih dari lima orang memiliki akses tulis ke lingkungan produksi?
4. Bagilah skor Anda dengan 2. Ini adalah CI/CD skor pipeline Anda.
5. Bandingkan CI/CD skor pipeline Anda dengan tabel berikut untuk menentukan CI/CD level pipeline Anda.

Skor CI/CD	Tingkat CI/CD
9.5 atau di atas	Sepenuhnya CI/CD
8—9	Sebagian besar CI/CD
5—7	Netral
Di bawah 5	Bukan CI/CD

Jika Anda mencetak gol di bawah 8, kami sarankan Anda menetapkan tujuan untuk bergerak secara bertahap menuju level berikutnya. Ketika tujuan itu tercapai, maka pemangku kepentingan produk harus menilai apakah dan kapan tujuan baru harus ditetapkan. Tujuan dari latihan ini tidak selalu untuk mengadvokasi perubahan pada pipa Anda, melainkan untuk membawa kesadaran akan seperti

apa proses CI/CD penyebaran sepenuhnya dan di mana jaringan pipa Anda saat ini berada pada spektrum itu.

Praktik terbaik untuk CI/CD jaringan pipa

Berikut ini adalah praktik terbaik untuk CI/CD jaringan pipa sepenuhnya:

- Amankan lingkungan produksi — Karena memungkinkan untuk mencapai hampir semua yang diperlukan untuk pemeliharaan akun dan lingkungan dengan menggunakan IAc, penting untuk melakukan segala upaya untuk mengamankan lingkungan produksi dengan membatasi akses konsol dan program. Kami menyarankan untuk membatasi akses hanya ke beberapa, atau bahkan nol, pengguna. Saat Anda menerapkan IAC AWS CloudFormation, pengguna memerlukan izin terbatas. Sebagian besar izin diberikan ke CloudFormation layanan melalui peran layanan. Untuk informasi selengkapnya, lihat Peran layanan dalam CloudFormation dokumentasi dan Menerapkan kebijakan untuk izin hak istimewa paling sedikit. AWS CloudFormation
- Buat akun terpisah untuk setiap lingkungan — Dengan mendedikasikan akun terpisah untuk setiap lingkungan, Anda dapat menyederhanakan proses penyebaran dan membuat kontrol akses berbutir halus di tingkat akun. Ketika beberapa lingkungan berbagi sumber daya, itu mengurangi integritas lingkungan sebagai unit yang terisolasi. Yang terbaik adalah menjaga lingkungan tetap sinkron dan berbeda. Ini bahkan lebih penting bagi lingkungan produksi karena segala sesuatu dalam akun itu harus diperlakukan sebagai sumber daya produksi.
- Batasi informasi identitas pribadi (PII) ke lingkungan produksi - Baik untuk keamanan dan perlindungan dari risiko pertanggungjawaban, mengamankan PII sebanyak mungkin. Jika memungkinkan di lingkungan yang lebih rendah, gunakan data anonim atau sampel alih-alih menyalin data yang berpotensi sensitif dari lingkungan produksi.
- Tinjau kode dalam repositori — Proses CI/CD sepenuhnya mengurangi titik masuk untuk pipa ke satu titik, dan titik tunggal itu harus diamankan. Untuk alasan ini, Anda disarankan untuk memerlukan beberapa tinjauan kode sebelum menggabungkan cabang fitur ke cabang utama. Tinjauan kode ini dapat dilakukan oleh anggota tim yang memenuhi syarat, tetapi setidaknya satu anggota senior harus meninjau. Kode harus diuji secara ketat oleh pengulas. Bagaimanapun, cara terbaik untuk memperbaiki masalah dalam pipeline adalah dengan menghindari memasukkannya ke dalamnya. Juga, penting untuk menyelesaikan semua komentar yang dibuat oleh pengulas sebelum menggabungkan. Resolusi ini hanya bisa menjadi penjelasan mengapa tidak ada perubahan yang diperlukan, tetapi menangani semua komentar adalah pemeriksaan tambahan yang penting untuk membantu mencegah masuknya masalah ke dalam pipeline.
- Buat penggabungan kecil dan sering — Untuk memanfaatkan sepenuhnya integrasi berkelanjutan, ada baiknya untuk mendorong perubahan lokal ke dalam pipa secara terus menerus juga. Lagi

pula, jauh lebih bermanfaat bagi lingkungan pengembangan untuk tetap sinkron jika lingkungan lokal mengikutinya juga.

Untuk praktik terbaik lainnya untuk CI/CD jaringan pipa, lihat Ringkasan praktik terbaik dalam Mempraktikkan Integrasi Berkelanjutan dan Pengiriman Berkelanjutan di AWS.

Pertanyaan yang Sering Diajukan

Apa saja indikator utama bahwa proses penyebaran saya tidak sepenuhnya CI/CD?

Indikator yang paling umum adalah ketika ada beberapa cabang repositori yang mewakili lingkungan terpisah dalam pipa. Repositori dalam CI/CD proses penuh menggunakan alur kerja berbasis batang, di mana satu cabang bertindak sebagai sumber kebenaran tunggal untuk penerapan repositori itu. Untuk informasi selengkapnya, lihat Pendekatan berbasis batang. Indikator lain termasuk langkah-langkah penerapan manual selain keputusan go atau no-go sederhana, penggunaan hotfix, dan rilis terjadwal.

Bagaimana jika saya ingin menggunakan CI/CD proses sepenuhnya tetapi masih ingin menjadwalkan rilis fitur tertentu untuk titik waktu tertentu?

Ini biasanya dilakukan dengan flag fitur. Dalam proses ini, penerapan masih dilakukan terus menerus, tetapi fitur tertentu disembunyikan dengan menggunakan penutupan bersyarat dalam kode hingga tiba saatnya untuk merilisnya.

Bagaimana jika beberapa langkah dalam proses penerapan saya tidak dapat diotomatisasi?

Salah satu tujuan dari CI/CD pipa sepenuhnya adalah untuk meminimalkan kebutuhan akan proses manual, tetapi tentu saja ada kasus penggunaan potensial di mana proses manual mungkin diperlukan. Bahkan, proses read-only, seperti log aplikasi konsultasi, seringkali dapat dilakukan di lingkungan produksi dengan risiko minimal. Namun, sangat disarankan agar Anda memperlakukan tindakan penulisan manual dalam produksi sebagai upaya terakhir yang mutlak.

Bagaimana jika staf teknis saya lebih nyaman dengan alur kerja lama daripada dengan proses yang sepenuhnya? CI/CD

Adalah umum bagi staf teknis untuk tahan terhadap perubahan besar, terutama ketika sesuatu yang dulunya merupakan praktik terbaik digantikan oleh sesuatu yang lebih baru. Teknologi bergerak

cepat, dan peningkatan terus ditemukan. Meskipun tingkat skeptisisme tertentu adalah kualitas yang baik untuk dimiliki oleh staf teknis, sama pentingnya bagi mereka untuk terbuka terhadap perubahan. Jangan bergerak terlalu cepat dengan staf yang skeptis karena mereka perlu mengelola perubahan pada sistem sebelum diterapkan. Kuncinya adalah mencegah skeptis tetap statis selamanya.

Bagaimana jika lingkungan saya ada di beberapa akun? Bisakah saya masih menggunakan CI/CD proses sepenuhnya?

Ya, pada kenyataannya, disarankan untuk menggunakan akun terpisah untuk setiap lingkungan. Untuk informasi selengkapnya tentang pipeline yang mengaktifkan tahapan di akun yang berbeda, lihat Membuat pipeline CodePipeline yang menggunakan sumber daya dari akun lain Akun AWS.

Langkah berikutnya

Gunakan Uji lakmus untuk jaringan pipa CI/CD bagian ini untuk mengevaluasi DevOps proses dalam organisasi Anda. Tentukan apakah proses Anda sepenuhnya CI/CD. If they aren't, decide whether they need improvement to take full advantage of the benefits of CI/CD penerapan.

Bagaimana Anda tahu kapan Anda selesai? Jawabannya adalah bahwa banyak organisasi tidak pernah benar-benar selesai. Mereka berhenti di suatu tempat di sepanjang jalan, di tempat yang cocok untuk kasus penggunaan mereka. Meskipun CI/CD jalur pipa sepenuhnya adalah skenario kasus terbaik, itu sangat tergantung pada situasi organisasi dan pemangku kepentingan di balik keputusan tersebut. Pemangku kepentingan harus memutuskan tahap CI/CD implementasi mana yang paling sesuai untuk kasus penggunaan mereka dan cara terbaik untuk memetakan perkembangan ke fase berikutnya.

Untuk informasi lebih lanjut tentang merancang dan membangun CI/CD jaringan pipa, lihatSumber daya.

Sumber daya

AWS dokumentasi dan referensi

- Arsitektur Referensi Pipeline Deployment AWS
- Apa itu Pengiriman Berkelanjutan?
- Mempraktikkan Integrasi Berkelanjutan dan Pengiriman Berkelanjutan di AWS (AWS Whitepaper)
- Siapkan CI/CD Pipeline di AWS (AWS Tutorial Langsung)
- Buat pipeline Wilayah AWS yang tidak mendukung AWS CodePipeline (Panduan AWS Preskriptif)
- Arsitektur Referensi Pipelines Penerapan dan Implementasi Referensi (posting blog)AWS

Layanan dan alat

- Tes CI/CD Lakmus
- AWS Cloud Development Kit (AWS CDK)
- AWS CloudFormation
- AWS CodePipeline

Riwayat dokumen

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

Perubahan	Deskripsi	Tanggal
Publikasi awal	—	Agustus 25, 2023

AWS Glosarium Panduan Preskriptif

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

Nomor

7 Rs

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

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

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

A

ABAC

Lihat kontrol akses berbasis atribut.

layanan abstrak

Lihat layanan terkelola.

ASAM

Lihat atomisitas, konsistensi, isolasi, daya tahan.

migrasi aktif-aktif

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

migrasi aktif-pasif

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

fungsi agregat

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

AI

Lihat kecerdasan buatan.

AIOps

Lihat operasi kecerdasan buatan.

anonimisasi

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

anti-pola

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

kontrol aplikasi

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

portofolio aplikasi

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

kecerdasan buatan (AI)

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

operasi kecerdasan buatan (AIOps)

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

enkripsi asimetris

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

atomisitas, konsistensi, isolasi, daya tahan (ACID)

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

kontrol akses berbasis atribut (ABAC)

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

sumber data otoritatif

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

Zona Ketersediaan

Lokasi berbeda di dalam 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 Region, atau di seluruh organisasi, dengan menggunakan templat YAMM. Untuk informasi selengkapnya, lihat Paket kesesuaian dalam dokumentasi. AWS Config

integrasi berkelanjutan dan pengiriman berkelanjutan (CI/CD)

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

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

CV

Lihat visi komputer.

D

data saat istirahat

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

klasifikasi data

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

penyimpangan data

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

data dalam transit

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

jala data

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

minimalisasi data

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

perimeter data

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

prapemrosesan data

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

asal data

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

subjek data

Individu yang datanya dikumpulkan dan diproses.

gudang data

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

bahasa definisi database (DDL)

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

bahasa manipulasi basis data (DHTML)

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

DDL

Lihat bahasa definisi database.

ansambel yang dalam

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

pembelajaran mendalam

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

defense-in-depth

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

administrator yang didelegasikan

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

deployment

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

lingkungan pengembangan

Lihat lingkungan.

kontrol detektif

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

pemetaan aliran nilai pengembangan (DVSM)

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

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

kembar digital

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

tabel dimensi

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

musibah

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

pemulihan bencana (DR)

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

DML~

Lihat bahasa manipulasi basis data.

desain berbasis domain

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

DR

Lihat pemulihan bencana.

deteksi drift

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

DVSM

Lihat pemetaan aliran nilai pengembangan.

E

EDA

Lihat analisis data eksplorasi.

EDI

Lihat pertukaran data elektronik.

komputasi tepi

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

pertukaran data elektronik (EDI)

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

enkripsi

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

kunci enkripsi

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

endianness

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

titik akhir

Lihat titik akhir layanan.

layanan endpoint

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

perencanaan sumber daya perusahaan (ERP)

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

enkripsi amplop

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

lingkungan

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

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

epik

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

ERP

Lihat perencanaan sumber daya perusahaan.

analisis data eksplorasi (EDA)

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

F

tabel fakta

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

gagal cepat

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

batas isolasi kesalahan

Dalam AWS Cloud, batas seperti Availability Zone, 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 bidikan nol.

FGAC

Lihat kontrol akses berbutir halus.

kontrol akses berbutir halus (FGAC)

Penggunaan beberapa kondisi untuk mengizinkan atau menolak permintaan akses.

migrasi flash-cut

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

FM

Lihat model pondasi.

model pondasi (FM)

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

G

AI generatif

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

pemblokiran geografis

Lihat pembatasan geografis.

pembatasan geografis (pemblokiran geografis)

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

Alur kerja Gitflow

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

gambar emas

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

strategi greenfield

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

pagar pembatas

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

H

HA

Lihat ketersediaan tinggi.

migrasi database heterogen

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

ketersediaan tinggi (HA)

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

modernisasi sejarawan

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

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

data penahanan

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

migrasi database homogen

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

data panas

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

perbaikan terbaru

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

periode hypercare

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

I

IAc

Lihat infrastruktur sebagai kode.

kebijakan berbasis identitas

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

aplikasi idle

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

IIoT

Lihat Internet of Things industri.

infrastruktur yang tidak dapat diubah

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

masuk (masuknya) VPC

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

migrasi inkremental

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

Industri 4.0

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

infrastruktur

Semua sumber daya dan aset yang terkandung dalam lingkungan aplikasi.

infrastruktur sebagai kode (IAc)

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

Internet of Things industri (IIoT)

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

inspeksi VPC

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

Internet of Things (IoT)

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

interpretabilitas

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

IoT

Lihat Internet of Things.

Perpustakaan informasi TI (ITIL)

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

Manajemen layanan TI (ITSM)

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

ITIL

Lihat perpustakaan informasi TI.

ITSM

Lihat manajemen layanan TI.

L

kontrol akses berbasis label (LBAC)

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

landing zone

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

model bahasa besar (LLM)

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

migrasi besar

Migrasi 300 atau lebih server.

LBAC

Lihat kontrol akses berbasis label.

hak istimewa paling sedikit

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

angkat dan geser

Lihat 7 Rs.

sistem endian kecil

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

LLM

Lihat model bahasa besar.

lingkungan yang lebih rendah

Lihat lingkungan.

M

pembelajaran mesin (ML)

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

cabang utama

Lihat cabang.

malware

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

layanan terkelola

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

sistem eksekusi manufaktur (MES)

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

PETA

Lihat Program Percepatan Migrasi.

mekanisme

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

akun anggota

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

MES

Lihat sistem eksekusi manufaktur.

Transportasi Telemetri Antrian Pesan (MQTT)

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

layanan mikro

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

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

arsitektur microservices

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

Program Percepatan Migrasi (MAP)

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

migrasi dalam skala

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

pabrik migrasi

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

metadata migrasi

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

pola migrasi

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

Penilaian Portofolio Migrasi (MPA)

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

Penilaian Kesiapan Migrasi (MRA)

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

strategi migrasi

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

ML

Lihat pembelajaran mesin.

modernisasi

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

penilaian kesiapan modernisasi

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

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

aplikasi monolitik (monolit)

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

MPA

Lihat Penilaian Portofolio Migrasi.

MQTT

Lihat Transportasi Telemetri Antrian Pesan.

klasifikasi multiclass

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

infrastruktur yang bisa berubah

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

O

OAC

Lihat kontrol akses asal.

OAI

Lihat identitas akses asal.

OCM

Lihat manajemen perubahan organisasi.

migrasi offline

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

OI

Lihat integrasi operasi.

OLA

Lihat perjanjian tingkat operasional.

migrasi online

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

OPC-UA

Lihat Komunikasi Proses Terbuka - Arsitektur Terpadu.

Komunikasi Proses Terbuka - Arsitektur Terpadu (OPC-UA)

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

perjanjian tingkat operasional (OLA)

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

Tinjauan Kesiapan Operasional (ORR)

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

teknologi operasional (OT)

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

integrasi operasi (OI)

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

jejak organisasi

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

manajemen perubahan organisasi (OCM)

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

kontrol akses asal (OAC)

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

identitas akses asal (OAI)

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

ORR

Lihat tinjauan kesiapan operasional.

OT

Lihat teknologi operasional.

keluar (jalan keluar) VPC

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

P

batas izin

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

Informasi Identifikasi Pribadi (PII)

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

PII

Lihat informasi yang dapat diidentifikasi secara pribadi.

buku pedoman

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

PLC

Lihat pengontrol logika yang dapat diprogram.

PLM

Lihat manajemen siklus hidup produk.

kebijakan

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

ketekunan poliglot

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

penilaian portofolio

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

predikat

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

predikat pushdown

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

kontrol preventif

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

principal

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

privasi berdasarkan desain

Pendekatan rekayasa sistem yang memperhitungkan privasi melalui seluruh proses pengembangan.

zona yang dihosting pribadi

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

kontrol proaktif

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

manajemen siklus hidup produk (PLM)

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

lingkungan produksi

Lihat lingkungan.

pengontrol logika yang dapat diprogram (PLC)

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

rantai cepat

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

pseudonimisasi

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

publish/subscribe (pub/sub)

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

Q

rencana kueri

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

regresi rencana kueri

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

R

Matriks RACI

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

LAP

Lihat Retrieval Augmented Generation.

ransomware

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

Matriks RASCI

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

RCAC

Lihat kontrol akses baris dan kolom.

replika baca

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

arsitek ulang

Lihat 7 Rs.

tujuan titik pemulihan (RPO)

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

tujuan waktu pemulihan (RTO)

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

refactor

Lihat 7 Rs.

Region

Kumpulan AWS sumber daya di wilayah geografis. Masing-masing 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). Tipe dukungan (S) adalah opsional. Jika Anda menyertakan dukungan, matriks disebut matriks RASCI, dan jika Anda mengecualikannya, itu disebut matriks RACI.

kontrol responsif

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

melestarikan

Lihat 7 Rs.

pensiun

Lihat 7 Rs.

Retrieval Augmented Generation (RAG)

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

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

rotasi

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

kontrol akses baris dan kolom (RCAC)

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

RPO

Lihat tujuan titik pemulihan.

RTO

Lihat tujuan waktu pemulihan.

buku runbook

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

D

SAML 2.0

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

SCADA

Lihat kontrol pengawasan dan akuisisi data.

SCP

Lihat kebijakan kontrol layanan.

Rahasia

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

keamanan dengan desain

Pendekatan rekayasa sistem yang memperhitungkan keamanan melalui seluruh proses pengembangan.

kontrol keamanan

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

pengerasan keamanan

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

sistem informasi keamanan dan manajemen acara (SIEM)

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

otomatisasi respons keamanan

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

enkripsi sisi server

Enkripsi data di tujuannya, oleh Layanan AWS yang menerimanya.

kebijakan kontrol layanan (SCP)

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

titik akhir layanan

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

perjanjian tingkat layanan (SLA)

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

indikator tingkat layanan (SLI)

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

tujuan tingkat layanan (SLO)

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

model tanggung jawab bersama

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

SIEM

Lihat informasi keamanan dan sistem manajemen acara.

titik kegagalan tunggal (SPOF)

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

SLA

Lihat perjanjian tingkat layanan.

SLI

Lihat indikator tingkat layanan.

SLO

Lihat tujuan tingkat layanan.

split-and-seed model

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

SPOF

Lihat satu titik kegagalan.

skema bintang

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

pola ara pencekik

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

subnet

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

kontrol pengawasan dan akuisisi data (SCADA)

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

enkripsi simetris

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

pengujian sintetis

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

sistem prompt

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

T

tag

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

variabel target

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

daftar tugas

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

lingkungan uji

Lihat lingkungan.

pelatihan

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

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

gerbang transit

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

alur kerja berbasis batang

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

akses tepercaya

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

penyetelan

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

tim dua pizza

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

U

waswas

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

ketidakpastian aleatorik disebabkan oleh kebisingan dan keacakan yang melekat dalam data. Untuk informasi lebih lanjut, lihat panduan Mengukur ketidakpastian dalam sistem pembelajaran mendalam.

tugas yang tidak terdiferensiasi

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

lingkungan atas

Lihat lingkungan.

V

menyedot debu

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

kendali versi

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

Peering VPC

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

kerentanan

Kelemahan perangkat lunak atau perangkat keras yang membahayakan keamanan sistem.

W

cache hangat

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

data hangat

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

fungsi jendela

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

beban kerja

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

aliran kerja

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

CACING

Lihat menulis sekali, baca banyak.

WQF

Lihat AWS Kerangka Kualifikasi Beban Kerja.

tulis sekali, baca banyak (WORM)

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

Z

eksploitasi zero-day

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

kerentanan zero-day

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

bisikan zero-shot

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

aplikasi zombie

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

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