Engineer

Bygde CI/CD-pipelines i GitHub Actions

Bygde CI/CD-pipelines i GitHub Actions med et distroless-produksjonsimage for frontend og promotering gjennom flere miljøer.

2025

én pipeline, tre miljøer, ett distroless-image

Situasjon. Leveranser bør ikke avhenge av at noen husker trinnene, og det som til slutt kjører i produksjon, bør ikke være en tung container for generell bruk med et skall og en pakkebehandler den aldri kommer til å bruke — det er bare angrepsflate som ligger der uten grunn.

Oppgave. Gjør veien fra commit til kjøring i Azure automatisk, og hold produksjonsimagene så små og låst ned som hver arbeidslast tillater.

Handling. Leveransepipelinen er GitHub Actions. Egne arbeidsflyter håndterer kvalitetsporten for koden, testene og produksjonssettingen til hvert miljø, med CodeQL, avhengighetsgjennomgang og et SBOM-steg ved siden av, slik at ingenting når et miljø uten å ha bestått kontrollene først. Imagene er flertrinnsbygg, og basen for hver del ble valgt ut fra hva den delen trenger, i stedet for som ett felles valg: frontenden leveres på et distroless-image (gcr.io/distroless/cc-debian13 — uten skall, uten pakkebehandler), Go-API-et på en slank Alpine og databaseimaget på postgres-slim. Promotering flytter et bygg gjennom miljøene langs en fast rute, ikke manuelt.

Resultat. Utgivelser sluttet å være et forsiktig manuelt ritual og ble en rutinemessig, kjedelig hendelse, som er nøyaktig det man vil ha fra utgivelser. Produksjonsfrontenden kjører på omtrent så lite som det går an å gi den, kontrollene fanger problemer før de lander, og «produksjonssetting» er noe leveransepipelinen gjør, ikke noe noen sveiter seg gjennom.

Bruk erfaringen i prosjektet deres

Bruk «Bygde CI/CD-pipelines i GitHub Actions» som utgangspunkt for en lignende utfordring i organisasjonen deres. Tjenestene bak arbeidet er tilgjengelige for prosjektet deres. Testet, utgitt og mulig å reversere, slik at endringer når kundene en tirsdag ettermiddag i stedet for sent en fredagskveld.

Se tjenestene våre Ta kontakt →

Se bak resultatet

Les om situasjonen, beslutningene og resultatet i eksemplet. Vi kan drøfte hva som passer deres rammer, og hva som krever en annen tilnærming. Bygde CI/CD-pipelines i GitHub Actions — én pipeline, tre miljøer, ett distroless-image

Se tjenestene våre Ta kontakt →

Trenger du en ordentlig CI/CD-pipeline?

Snakk om prosjektet med ingeniøren som har ansvaret for leveransen. Vi avtaler omfang, prisgrunnlag og overlevering skriftlig før arbeidet begynner.

Rolle i oppdraget: DevOps-ingeniør. Det bygger på DevOps, Skytjenester, Containere (Docker/Kubernetes), Automatisering og CI/CD, Pålitelighet og sikkerhetskopier, Sikkerhet og Testing og kvalitetssikring. Og nå som dere er her: vi er ganske gode på DevOps og CI/CD-automatisering og Containerisering og orkestrering.

Ta kontakt Se tjenestene våre

Ett stykke av **en lengre oversikt** — og den ligger i sin helhet her.

Hver oppføring er skrevet på samme måte: situasjonen, oppgaven, hva vi gjorde, og hva som endret seg.

Se hele oversikten

Definer forbedringen dere trenger

Ta med utrullingsløpet, driftsrisikoene og dem som har vakt. Vi kan avgrense en endring med kontroller, en vei tilbake og tydelig ansvar.

Se hva det koster Ta kontakt →

Trenger du en ordentlig CI/CD-pipeline?

Spør om beslutningene bak «Bygde CI/CD-pipelines i GitHub Actions». Dere snakker med folk som kan forklare arbeidet og vurdere deres eget prosjekt.

Se tjenestene våre

welcome@engineer.company +45 50 10 61 84 København, Danmark

Ta med deg denne siden

Visittkort

Laget med kjærlighet av D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

Opphavsrett © 2025 – nå · Engineer ApS · Alle rettigheter forbeholdt.

Skrevet ut fra https://engineer.company/nb/portfolio/built-github-actions-ci-cd-pipelines/