

Einsatz von DevOps und containerbasierten CI/CD Pipelines am Beispiel der Datenintegrationsplattform



Stefanie Lehn – Datenintegrationszentrum Gießen

Gliederung

- Datenintegrationsplattform
- DevOps
- Continuous Integration, Continuous Delivery, Continuous Deployment
- GitLab CI
- Beispiel Pipeline
- Best Practices
- Vorteile/Nachteile
- Zusammenfassung

Datenintegrationsplattform (DIP)

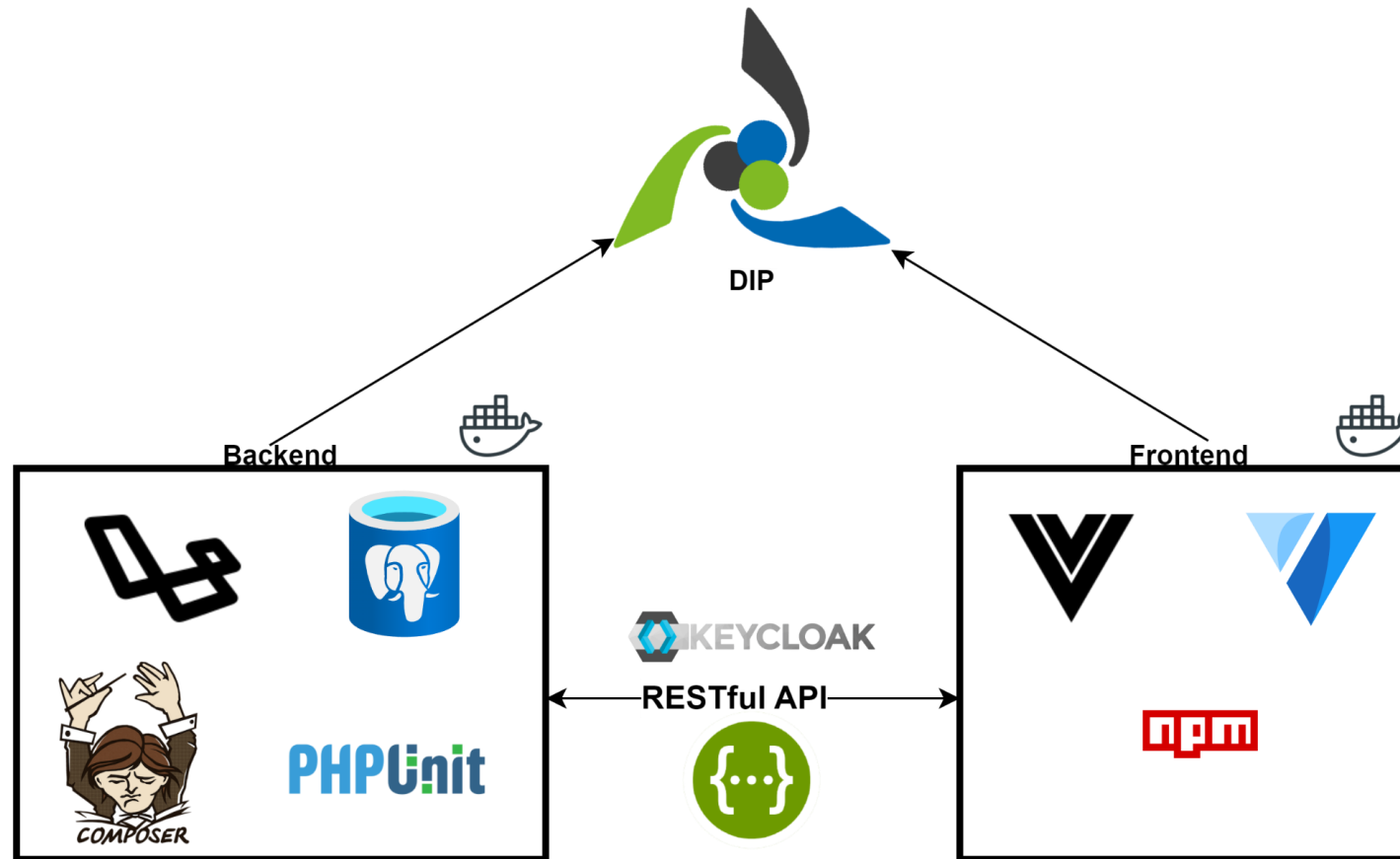
Datenmanagement mithilfe
von **Datensätzen** und
Werkzeugen



Tools zur Unterstützung der
Forschungstätigkeiten sind in
die DIP **integriert**

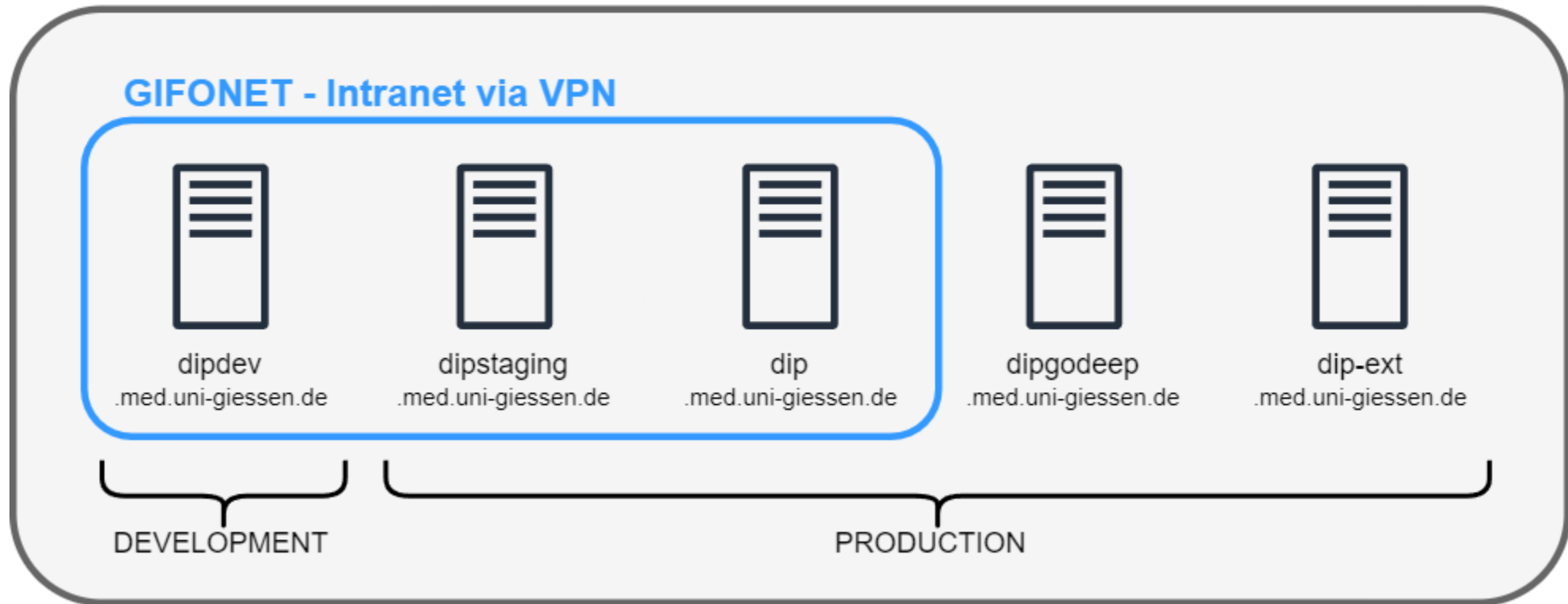
- Fallzahlabfrage
- Forschungsanträge stellen
- Analysesoftware
 - OMOP
 - i2b2
 - FHIR
 - RStudio
 - Dashboards

Datenintegrationsplattform (DIP)



Datenintegrationsplattform (DIP)

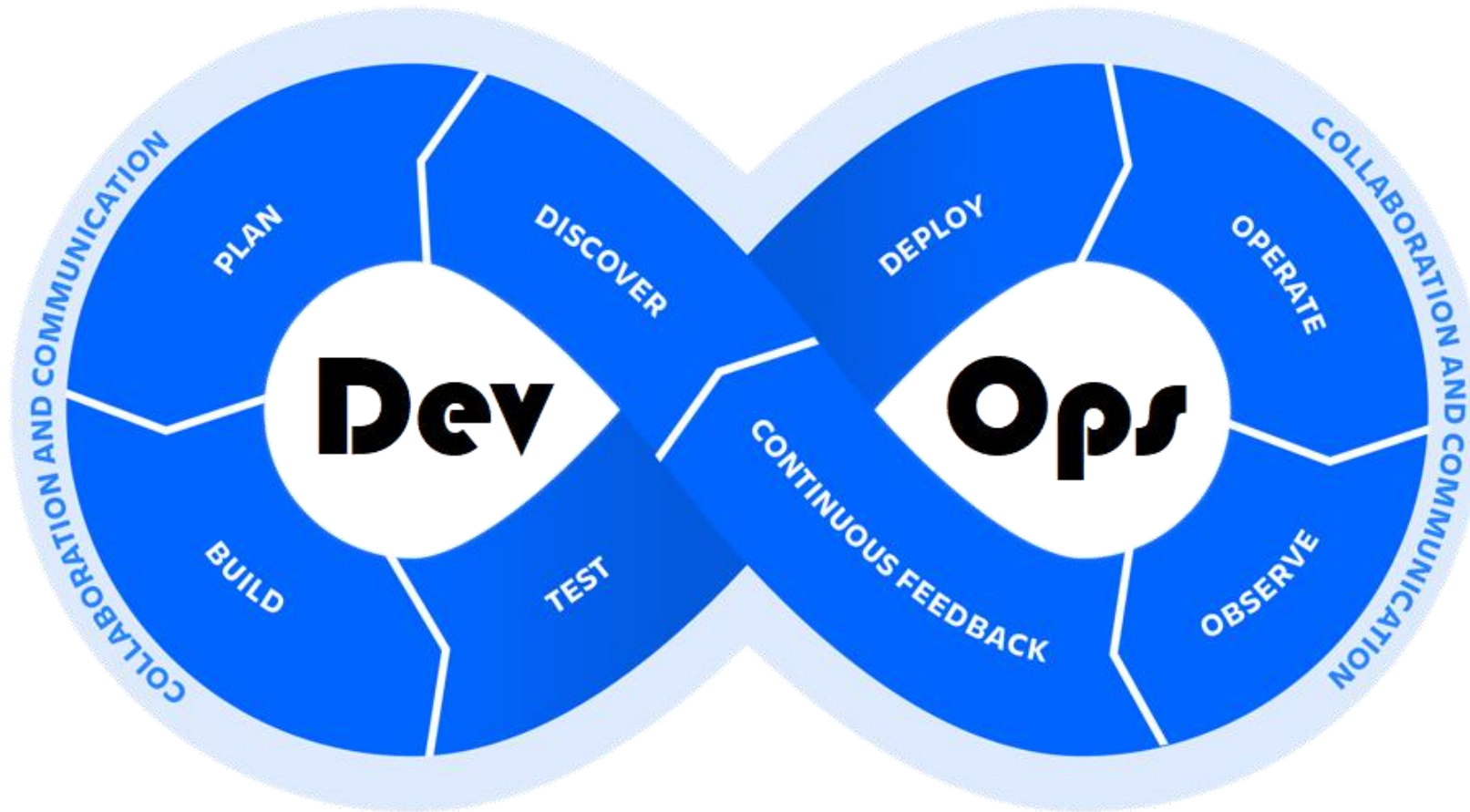
Deployment Virtual Machines



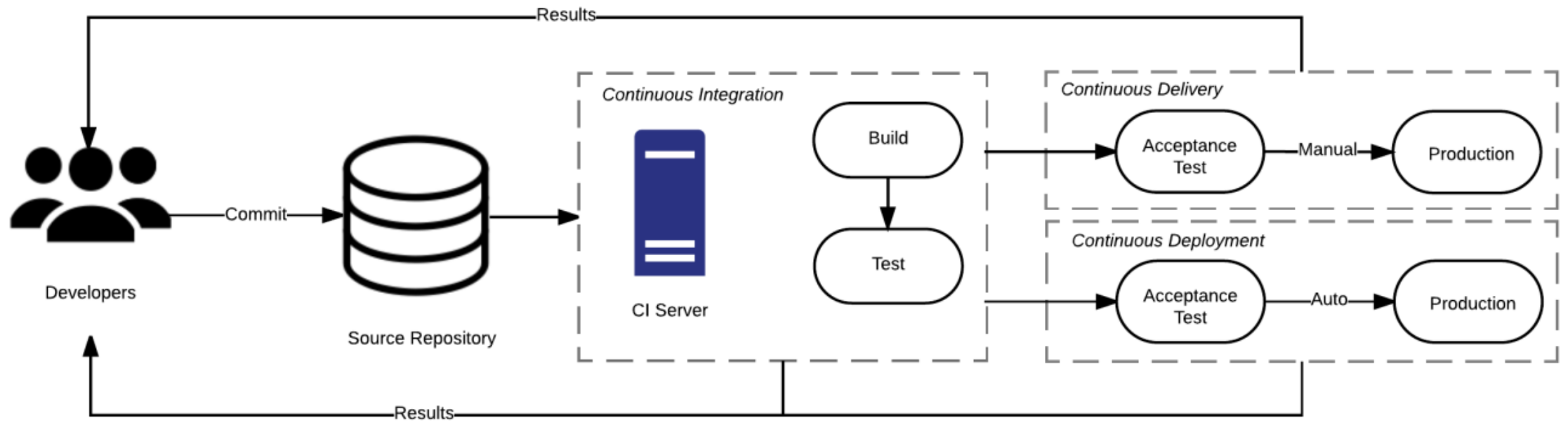
DevOps

- Basierend auf Agile Prinzipien
- Interdisziplinäre Teams für Development und Operations
- Effektive Zusammenarbeit und Kommunikation
- Kundenfeedback und Kundeninteresse im Mittelpunkt
- Monitoring und Metriken zur Überwachung von Systemen und Identifizierung von Optimierungspotenzial
- **Automatisierung**

DevOps





Continuous Integration, -Delivery, -Deployment

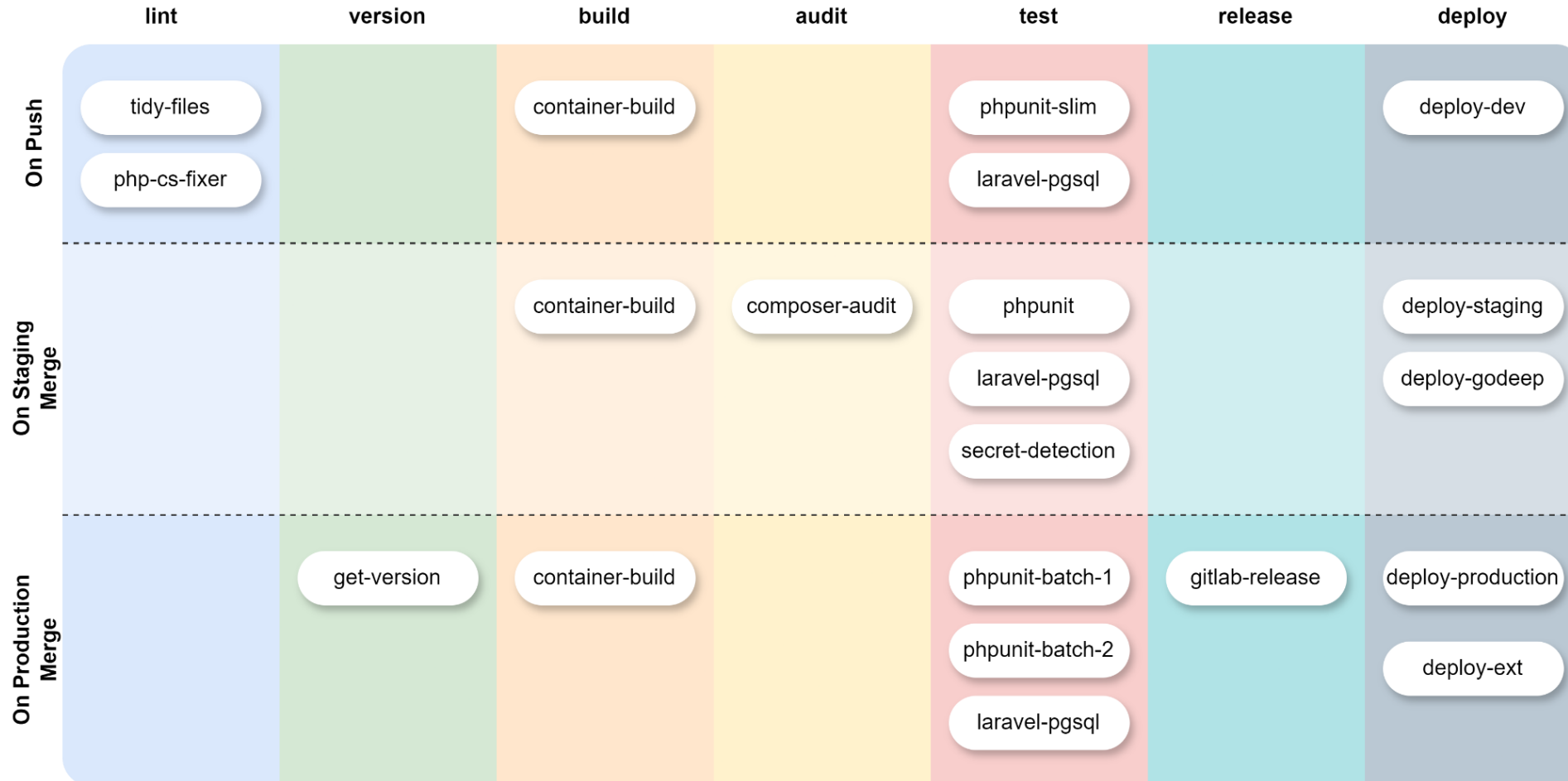


GitLab CI

- Definition von Pipeline stages und jobs in **gitlab-ci.yml** Datei
- GitLab **Runner** zur Ausführung der definierten Jobs
- GitLab **Container Registry**

<div>✓ Passed</div> <div>🕒 00:02:11</div> <div>📅 1 month ago</div>	<div>ci: changed rules for secret detection</div> <div>#6649 🔗 670 🔑 10d8b372 🐙</div> <div>merge request</div>		<div>✓✓✓✓✓</div>	<div>📄 ▼</div>
<div>✗ Failed</div> <div>🕒 00:01:58</div> <div>📅 1 month ago</div>	<div>ci: changed trivy command</div> <div>#6639</div> <div>🔗 910-ci-try-out-container-scanning-in-pipeline</div> <div>🔑 8df869dc 🐙</div> <div>latest</div>		<div>✓✓✓✗»🔧</div>	<div>🔄 📄 ▼</div>

DIP Backend CI/CD Pipeline



DIP Build Job

container-build:

rules:

- if: "\$CI_MERGE_REQUEST_ID && \$CI_MERGE_REQUEST_TARGET_BRANCH_NAME == \$CI_DEFAULT_BRANCH"
- if: \$CI_PIPELINE_SOURCE == "push" && \$CI_COMMIT_BRANCH == \$CI_DEFAULT_BRANCH
- if: \$CI_PIPELINE_SOURCE == "push" && \$CI_COMMIT_BRANCH == \$CI_RELEASE_BRANCH

stage: build

image:

name: gcr.io/kaniko-project/executor:v1.23.2-debug

entrypoint:

- ''

script:

```
/kaniko/executor \
--context "${CI_PROJECT_DIR}" \
--dockerfile "${CI_PROJECT_DIR}/Dockerfile" \
--destination "$IMAGE_PATH:$CI_COMMIT_SHORT_SHA" \
--destination "$IMAGE_PATH:$CI_COMMIT_REF_SLUG" \
--cache=true \
--cache-repo="${IMAGE_PATH}/cache_image" \
--cache-ttl=72h;
```

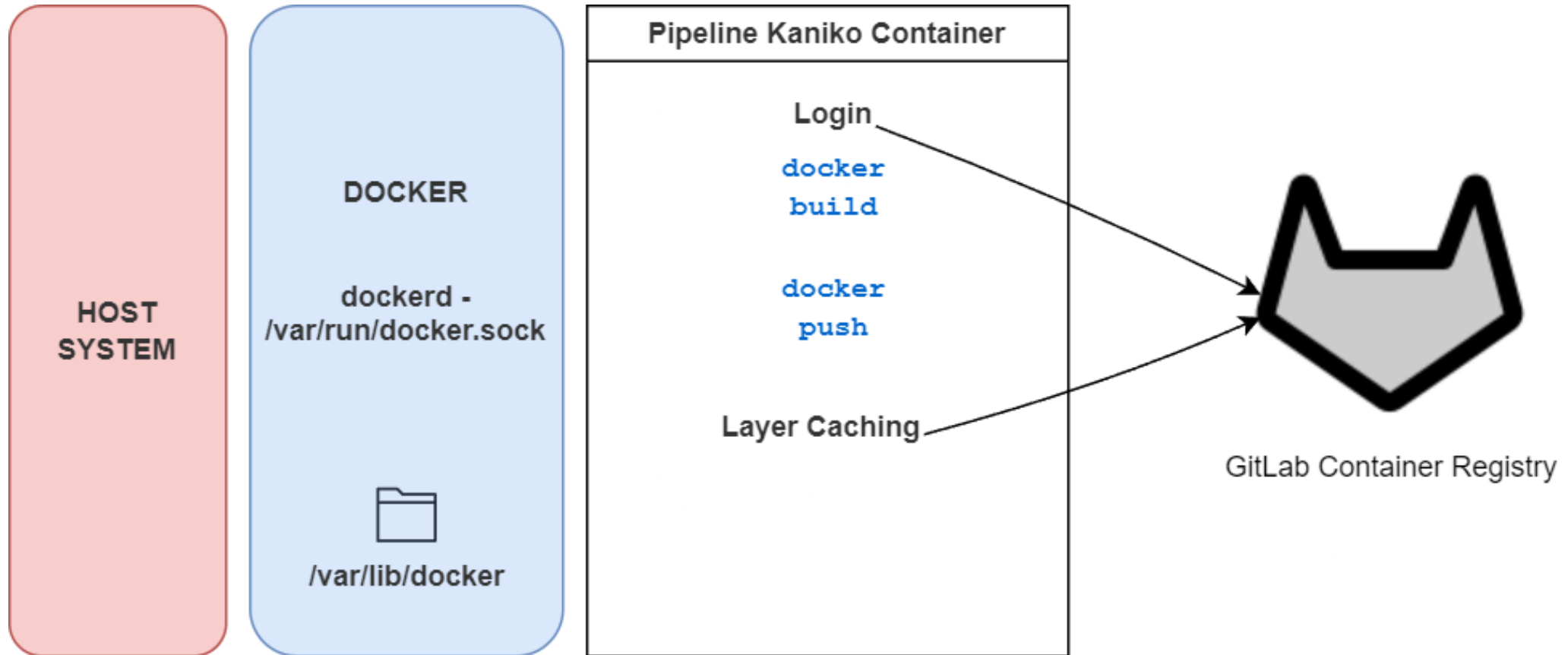
extends:

- ".run_always"

tags:

- docker

DIP Build Job



Best Practices

- Grundlegende CI/CD Komponenten:
 - Version Control
 - Build Automatisierung
 - Test Automatisierung
 - Automatisiertes Deployment
- Performancekriterien:
 - Dockerfiles effizient strukturieren, Build Tool + Caching, Testoptimierung, Entfernen von Redundanzen
- Sicherheitskriterien:
 - Dependencies überprüfen/auditieren, Docker Images überprüfen, Secret Detection

Vorteile/Nachteile

- + Kontinuierliche Verifizierung
 - + Automatisierung
 - + Kontinuierliches Feedback
 - + Höhere Codequalität
 - + Mehr Kontrolle
-
- Implementierungsaufwand (Infrastruktur, Personal, Tools)
 - Dokumentation nötig

Zusammenfassung

- DevOps verbessert Zusammenarbeit zwischen Development und Operations für verlässlichere, risikoärmere Software Releases
- Automatisierung in Form von Continuous Integration, Continuous Delivery, Continuous Deployment
- GitLab CI integriert CI/CD/CD in Versionskontrolle
- Pipelinedefinition in gitlab-ci.yml Datei, containerisierte Ausführung durch GitLab Runner
- Grundlegende CI/CD Jobs: Build, Test, Deployment (vereinfacht durch Verwendung von Containern)
- Optimierungen beim Software Build und Test für bessere Performance
- Sicherheitsrelevante Aspekte wie dependency audits und container image scans integrieren
- DevOps & CI/CD/CD verbessern Qualität des Softwareprodukts, die Etablierung ist aber mit Aufwand verbunden

Fragen + Antworten

- Testing im Frontend nicht automatisiert? Unterbrechung der Pipeline?
 - Akzeptanztests findet aktuell (noch) manuell statt, nachdem die Änderungen an der Software auf einem Testsystem deployed wurden.
- Dokumentation als wichtiger Punkt erwähnt. Wo wird die Pipeline dokumentiert?
 - Pipelinedokumentation im GitLab Wiki zusammen mit Contribution Guides und Softwaredokumentation.
- Hürden bei Etablierung von DevOps und CI/CD/CD?
 - Erfolg hängt stark von Organisation, Agilität/Flexibilität im Projektmanagement und der Qualität der Kommunikation ab. DevOps baut auf agilen Ansätzen auf und benötigt eventuell auch einen unternehmenskulturellen Wandel, sowie effektive Kommunikation.

DatenIntegrationsZentrum Gießen

Vernetzen. Forschen. Heilen

Kontakt:

Datenintegrationszentrum Gießen

Rudolf-Buchheim-Str. 6
35392 Gießen

