

teuto.net Konzept verein.cloud

Burkhard Noltensmeier, Jun 2021

Contents

1	verein.cloud Projekt der Tineon AG	2
1.1	IOS & Android App incl. NodeJS Backend	2
1.2	Angular Frontend	2
2	Philosophie von teuto.net	3
3	Agile Entwicklung	3
4	Verantwortlichkeiten der Developer Teams (aus Sicht von teuto.net)	5
5	Verantwortlichkeiten Tineon (aus Sicht von teuto.net)	5
6	Verantwortlichkeiten teuto.net	5
6.1	24x7 Betrieb der Infrastruktur	5
6.2	24x7 Betrieb Managed Kubernetes Service inklusive:	5
6.3	24x7 Betrieb Monitoring	5
6.4	24x7 Betrieb Managed GitLab als zentrale Source Code und Deployment Platform.	6
6.5	CI/CD Pipline	6
7	SLA	6
8	Timeline aus Sicht von teuto.net	7

1 verein.cloud Projekt der Tineon AG

Wir wiederholen hier noch einmal Stichpunktartig wie wir das Projekt verstehen:

- Tineon betreibt eine Vereinsverwaltungssoftware.
- Diese Software verfügt über eine Webschnittstelle und eine Benutzerverwaltung.
- Die Software wird aktuell “klassisch” bei Hetzner gehostet.

1.1 IOS & Android App incl. NodeJS Backend

- Tineon hat ein nearshoring Unternehmen im Kosovo mit der Entwicklung eine Android- und IOS-App inklusive Backend beauftragt.
- Das Backend besteht aus einem NodeJS Server sowie einer MYSQL Datenbank und eine Keycloak Authentication Server.
- User der bestehenden Vereinverwaltungssoftware werden automatisch auf dem Keycloak registriert.
- Die Apps nutzen zusätzlich den Firebase Server von Google, um einen Chat bereitzustellen.
- Die Steuerung der Entwicklungsprozesse wird mittels Jira Tickets durchgeführt.
- Die Entwicklung Anwendung hat einen Stand erreicht mit dem diese in Produktion genommen werden kann.
- Unit Tests oder Integrationstests für die Anwendung sind nicht vorhanden.
- Perspektivisch soll die Anwendung Dokumente per S3 Schnittstelle im Backend verwalten.

1.2 Angular Frontend

- Mit der Entwicklung eines Angular Frontend wurde ein Unternehmen in Indien beauftragt.
- Diese Unternehmen hat eine Kopie des Backend Source Codes bekommen und ist mit der Entwicklung gestartet.

2 Philosophie von teuto.net

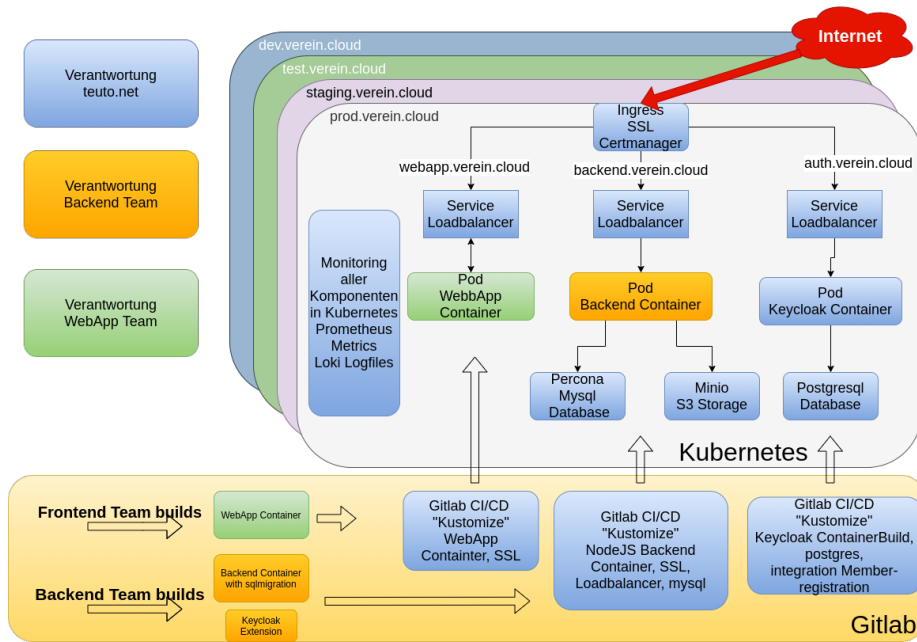
teuto.net

- schult und organisiert DevOPS Entwicklungsprozesse für Unternehmen.
- ist Mitglied der Cloud Native Foundation und beteiligt sich an der Entwicklung von Opensource Projekten.
- automatisiert Continuous Integration und Deployment mit Git bzw. Gitlab Pipelines.
- betreibt managed Kubernetes Cluster und stellt Ressourcen wie Containerbetrieb, Storage, Networking, Loadbalancing und Networkfiltering zur Verfügung.
- verfügt über eigene Hardware zur Bereitstellung von Cloud Ressourcen (Datacenter Interxion Frankfurt)
- ist zertifizierter Kubernetes Service Provider
- verfügt über qualifizierte deutschsprachige “DevOPS engineers”
- fühlt sich der Transparenz aller Stakeholder und einem agilen Entwicklungsprozess verpflichtet

3 Agile Entwicklung

Wir schlagen agile Methoden für die Entwicklungs- und Deploymentprozesse vor:

- Alle Stakeholder arbeiten direkt auf den zentralen Gitlab Repositories (committen).
- Alle haben eine transparente Sicht auf den Code, die Pipelines, die Tickets und das Monitoring
- In den jeweiligen Repositories werden aus dem Code automatisch Docker Container erzeugt. (Angular WebApp bzw. des NodeJS Backends)
- Die Integration von neuen Features findet mittels Merge Request Workflow statt.
- Die Container können mit einer CI/CD Pipeline automatisch in eine dev|test|staging|prod Umgebung deployed werden



4 Verantwortlichkeiten der Developer Teams (aus Sicht von teuto.net)

- Koordinierung der verschiedenen Entwicklungs-/Feature Branches und deren Inhalte
- Softwaretests und Qualitätssicherung
- bauen und testen der Docker Container
- Für das Ausrollen der SQL Datenbank ist es notwendig, dass die Dienstleister das Datenmodell in automatisierbarer Form anliefern. (sqlmigration scripte).
- Auslösen des automatischen Deployments auf die jeweiligen Integrationssysteme dev|test|staging

5 Verantwortlichkeiten Tineon (aus Sicht von teuto.net)

- Releasemanagement
- Auslösen des automatischen Deployments auf das Produktionssystem

6 Verantwortlichkeiten teuto.net

6.1 24x7 Betrieb der Infrastruktur

- Internet inkl. Domain und DNS
- Storage inkl. offsite Backup
- virtuelle Maschinen

6.2 24x7 Betrieb Managed Kubernetes Service inklusive:

- Loadbalancer
- Percona Mysql Datenbankcluster
- S3 Storage (durch minio)
- Keycloak inkl. Postgresql Datenbank
- Automatische Bereitstellung SSL Zertifikate (Let's Encrypt)

6.3 24x7 Betrieb Monitoring

- 24x7 Monitoring des System (Grafana, Prometheus, Loki)
- Kapazitätsmanagement
- Überwachung der Systemkomponenten (konfiguration von spezifischen alarmen)

6.4 24x7 Betrieb Managed GitLab als zentrale Source Code und Deployment Platform.

- Installation und Betrieb eines Gitlabsystems
- Runner für Linux auf dem GitLab System
- Proaktives Monitoring
- OffSite Backup
- SSL mit Let's Encrypt
- Regelmäßige GitLab & Betriebssystem Updates

6.5 CI/CD Pipeline

- Entwickeln eine CI/CD Pipeline zum automatisierte Deployment in die Integrationsysteme
- Konfigurationsmanagement in Kubernetes für Loadblancer, SSL, Datenbanken, Keycloak
- Schulung und Support der Stakeholder im Entwicklungsprozess
- dauerhafter Betrieb und verbesserung der CI/CD Pipeline

7 SLA

- SLA 5x7
- Direkte und Qualifizierte Ansprechpartner
- Für den Regelbetrieb bieten wir ein Stunden-/Supportkontingent an aus dem alle beteiligten Stakeholder problemlos und effizient schöpfen können um anfallende Themen zu bearbeiten. Das Stundenkontingent ist erfahrungsgemäß mit 8h/Monat bei Erreichen des Regelbetriebs ausreichend dimensioniert.

8 **Timeline aus Sicht von teuto.net**

1. Vereinbarung eines hostnamen/domain für gitlab und testsystem
2. Bereitstellung des Gitlab System für Sourcecodeverwaltung, Dockerregistry und Deployment Pipelines
3. Konfiguration der Benutzer der einzelnen Teams und deren Projekten auf dem Gitlab System
4. Dienstleister übertragen den Sourcecodes in das gitlabssystem (nach Möglichkeit inkl. commit history)
5. Bereitstellung eines zentralen Test- und Integrationssystem auf Basis von Kubernetes
6. Erstellen von Dockercontainer für die Anwendung
7. Configuration Management der Anwendung auf Kubernetes umstellen
8. Entwickeln eines Kubernetes deployments für die Anwendung
9. Verfeinern der Deploymentpipeline
10. Weiterentwicklung der Anwendung in den entsprechenden Teams
11. Bereitstellung der Produktionsumgebung
12. Releasemanagement für das automatisierte Ausrollen der Anwendung in die Produktionsumgebung