Pflichtenheft

CMB Prod.

Auftraggeber: Johannes Lötzsch & Ina Rentzsch

Auftragnehmer: Eric Nözel, Emily Lehmann, Eka Yana, Marcel Risch & Jakob Vogel

Inhaltsverzeichnis

1. Zielsetzung des Projekts

Ziel des Projekts

Ziel ist die Erstellung einer stabilen und skalierbaren IT-Infrastruktur, die aus fünf virtuellen Maschinen (VMs) besteht. Die Infrastruktur soll:

- Webanfragen effizient verteilen und verarbeiten (Skalierbarkeit).
- Daten regelmäßig sichern (Datensicherheit).
- System Metriken kontinuierlich überwachen und visualisieren (Performance und Überwachung).

Anforderungen

- Aufbau von drei Core-VMs (Load Balancer und zwei Webserver).
- Zwei zusätzliche VMs für Backups und Monitoring.
- Automatisierte & regelmäßig laufende Backups.
- Monitoring mit Prometheus und Visualisierung in Grafana.
- Skalierbare Architektur mit Lastverteilung durch einen Load Balancer.
- Erfolgreiche Tests von Backups und Wiederherstellungen.

Zeitrahmen

Zeit für das gesamte Projekt: 28 UE

Projektmanagement (10 UE):

- Kickoff-Meeting: 4 UE
- Sprint-Planung und Backlog-Verfeinerung: 1 UE
- Daily Standups (5 Tage ca. 15 Minuten): 2 UE
- Sprint Review und Retrospektive: 1 UE
- Dokumentation (fortlaufend): 2 UE

Umsetzung (18UE):

- Ist dem Gantt-Diagramm zu entnehmen

2. Projektbeteiligte

- **Lenkungsausschuss**: Marcel Risch (Product Owner)
- **Projekt-Team**: Eric Nözel, Emily Lehmann, Eka Yana, Marcel Risch & Jakob Vogel
- **Stakeholder**:
    - Kunden & Consulting: Johannes Lötzsch & Rentsch
    - Projekt-Team
    - Product Owner
    - AFBB

3. Organisationsform

Klare Verantwortung & Fokussierung

Unser Team arbeitet ausschließlich an diesem Projekt und ist nicht in andere Abteilungen oder Tätigkeiten eingebunden. Dadurch können wir uns voll auf die Umsetzung der IT-Infrastruktur mit Webserver-Cluster, Monitoring und Backup konzentrieren.

- Alle Teammitglieder sind direkt dem Projekt zugeordnet.
- Es gibt eine klare Teamstruktur mit festen Rollen.

Autonomie & Entscheidungsfreiheit

Da wir eigenständig agieren und keine anderen Aufgaben außerhalb des Projekts haben, können wir schnelle Entscheidungen treffen. Wir sind nicht von anderen Abteilungen oder Vorgesetzten abhängig, was gerade in einem agilen Scrum-Prozess wichtig ist.

Beispiel: Wenn wir Anpassungen an der Infrastruktur oder Softwareauswahl machen müssen, brauchen wir keine Genehmigungen von externen Abteilungen – wir entscheiden direkt im Team.

4. Vorgehensmodelle

Scrum

Begründung

Scrum ist ein agiles Vorgehensmodell, das darauf ausgelegt ist, komplexe Projekte flexibel, iterativ und kundenorientiert zu entwickeln.

Von diesem Vorgehensmodell profitieren wir, weil …

1. Wir iterativ entwickeln können – Durch Sprints liefern wir regelmäßig funktionsfähige Ergebnisse.
2. Anforderungen flexibel bleiben – Änderungen können jederzeit berücksichtigt werden.
3. Unsere Teamstruktur klar definiert ist – Jeder hat feste Rollen und Verantwortlichkeiten.
4. Regelmäßiges Feedback den Fortschritt sichert – Probleme werden früh erkannt und behoben.
5. Transparenz und Zusammenarbeit im Team gefördert werden – Jeder kennt den aktuellen Stand.
6. Wir Unsicherheiten schrittweise bewältigen – Große Aufgaben werden in kleinere, lösbare Einheiten zerlegt.

Scrum-Rollen

Scrum-Master:	Emily Lehmann (Vertretung: Jakob Vogel)
Product-Owner:	Marcel Risch (Vertretung: Eka Yana)
Entwicklungsteam:	Eric Nözel
	Jakob Vogel
	Eka Yana

Pair Programming

Begründung

Pair Programming (Pair Configuring) ist eine agile Praxis, die die Zusammenarbeit verbessert und den Wissensaustausch fördert.

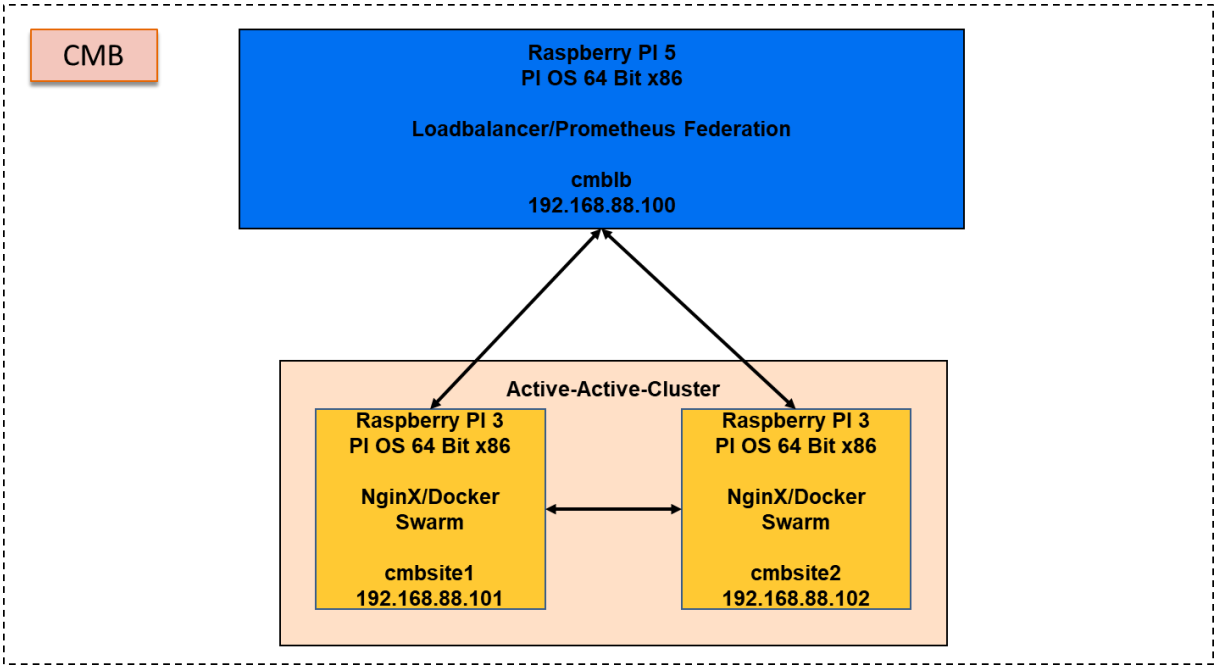
Von diesem Vorgehensmodell profitieren wir, weil …

1. Die Qualität der Konfiguration steigt – Fehler werden schneller erkannt und vermieden.
2. Wissen effektiver geteilt wird – Beide Beteiligten lernen und verstehen die Konfiguration besser.
3. Probleme gemeinsam gelöst werden – Zwei Perspektiven führen zu besseren Lösungen.
4. Die Teamarbeit verbessert wird – Kommunikation und Zusammenarbeit werden gefördert.
5. Verantwortlichkeiten klarer sind – Jeder kennt die Konfiguration und kann im Notfall übernehmen.

Pair-Programming-Rollen

Für den Webservercluster: Marcel Risch & Jakob Vogel

5. Anforderungen an die Infrastruktur



Übersicht der Komponenten

- **Core-VM1**: Load Balancer (Nginx)
- **Core-VM2 & Core-VM3**: Webserver (Docker-Container)
- **VM4**: Backup-Server (Rsync)
- **VM5**: Monitoring (Prometheus & Grafana)

Anforderungen an die einzelnen VMs

- **Core-VM1**:
  - Funktion: Verteilung der eingehenden Anfragen auf die Webserver
  - Technologie: Nginx als Load Balancer mit Least-Connection-Strategie

    - Anforderungen: Health Checks zur Erkennung defekter Webserver

- **Core-VM2 & Core-VM3**:
    - Funktion: Bereitstellung von Webinhalten
    - Technologie: Apache
    - Anforderungen: Redundanz und Lastverteilung durch identische Inhalte

- **VM4**:
    - Funktion: Automatisierte Sicherung der Daten
    - Technologie: Rsync
    - Anforderungen: Automatisierte Backups per Cronjobs

- **VM5**:
    - Funktion: Monitoring und Visualisierung
    - Technologie: Prometheus (Datensammlung) und Grafana (Visualisierung)
    - Anforderungen: Dashboards mit Echtzeitdaten und Verlaufsanalysen

Netzwerk-Anforderungen

- Interne Kommunikation zwischen den VMs
- Externe Verbindungen für Nutzeranfragen und API-Interaktionen
- Passwortgeschützte Netzwerkzugriffe durch SSH-Konfiguration

6. Funktionale Anforderungen

- **Load Balancer:** Muss Webanfragen intelligent auf die Webserver verteilen
- **Webserver:** Bereitstellung von Inhalten über eine dynamische Website
- **Monitoring:** Prometheus sammelt Systemmetriken, Grafana visualisiert sie
- **Backup:** Automatische Speicherung der Website-Konfiguration und Dokumentation zur Wiederherstellung

7. Nicht-funktionale Anforderungen

- **Performance**: Optimierte Ressourcennutzung auf allen VMs
- **Sicherheit**: SSH-Zugriff nur für autorisierte Benutzer, Backup-Verschlüsselung
- **Zuverlässigkeit**: Redundanz durch Webserver-Cluster und Load Balancing
- **Wartbarkeit**: Dokumentation aller Konfigurationen und Prozesse

Akzeptanzkriterien

- **Infrastruktur:** Alle VMs sind bereit und konfiguriert
- **Webserver:** Die Webseite läuft stabil in einem Docker-Container
- **Load Balancer:** Anfragen werden über Least-Connection verteilt
- **Monitoring:** Prometheus & Grafana erfassen und visualisieren Metriken
- **Backup:** Website-Konfiguration ist gesichert, Wiederherstellung ist dokumentiert

8. Risiken

Risiko	Wahrscheinlichkeit	Auswirkung	Maßnahmen

Fehlkonfiguration des Load Balancers	Mittel	Fehlfunktion der Webserver	Regelmäßige Tests und Backup der Konfigurationsdateien
Zeitliche Verzögerungen	Hoch	Nicht alle Features sind rechtzeitig fertig	Mindestanforderungen priorisieren, klare Sprintplanung
Unbekannte Softwareprobleme	Mittel	Zusätzlicher Arbeitsaufwand	Alternative Softwareoptionen prüfen, Troubleshooting-Dokumentation

9. Abnahmekriterien

1. Webserver mit Docker:
    - Ein Webserver ist eingerichtet, auf dem die Webseite in einem Docker-Container stabil läuft
    - Die Webseite ist über einen Browser erreichbar
2. Grundfunktionalität der Webseite:
    - Die Webseite muss grundlegende Inhalte anzeigen und ein Navigationsmenü besitzen.
    - Die Webseite ist responsiv und fehlerfrei aufrufbar
3. Monitoring mit Prometheus und Grafana:
    - Prometheus-Node-Exporter ist auf jeder virtuellen Maschine eingerichtet und funktionsfähig
    - Grafana zeigt mindestens eine benutzerdefinierte Metrik aus Prometheus an, z. B. die CPU-Auslastung eines Servers
4. Backup der Website-Konfiguration:
    - Die Website-Konfigurationsdateien sind im Backup enthalten
    - Eine Dokumentation beschreibt, wie das Backup eingespielt wird, um das System neu aufzusetzen

10. Zusätzliche Anforderung an das Produkt

- Erweiterte Monitoring-Funktionen:
    - Einrichtung von Grafana Dashboards zur Visualisierung von Metriken.
    - Alarme & Benachrichtigungen für kritische Systemzustände (z. B. per Slack oder E-Mail).
- Erhöhte Skalierbarkeit:
    - Zusätzliche Webserver-VMs für mehr Redundanz.
    - Automatische Skalierung je nach Last (z. B. mit Kubernetes oder Auto-Scaling).
- Sicherheitsverbesserungen:
    - Verschlüsselung der Backups zur Erhöhung der Datensicherheit.
    - Erweiterte Firewall-Regeln für mehr Schutz vor Angriffen.
- Automatisierung & DevOps-Ansätze:
    - CI/CD-Pipelines für automatisiertes Deployment.
    - Automatische Provisionierung mit Terraform oder Ansible.

11. Dokumentation und Abnahme

Erforderliche Dokumentationen

- Konfigurationsschritte für jede VM
- Backup- und Wiederherstellungsverfahren
- Monitoring-Dokumentation mit Dashboard-Übersicht
- Git Repository: https://github.com/WilderWilliGitHub/LF12_CMB-Prod

Abnahme: Erfolgt durch den **Product Owner**, wenn alle Akzeptanzkriterien erfüllt sind.