
# Konzeption – Entwicklung eines real-time Backends für eine datenintensive Applikation

**Seminararbeit**

Prüfungsleistung für den

**Master of Science**

des Studiengangs Data Science

an der Internationalen Hochschule

von

**Jasper Bremenkamp**

14. Oktober 2024

| | |
|---|---|
| **Kursbezeichnung** | Projekt: Data Engineering |
| **Matrikelnummer** | 92125193 |
| **Tutor** | Prof. Dr. Max Pumperla |

## 1 Szenario

Die Frischmarkt AG, eine Kette von Lebensmittelgeschäften, steht vor der Aufgabe, datengestützte Entscheidungsfindung in ihre Betriebsprozesse zu integrieren, um Effizienz zu steigern, Kosten zu senken und die Kundenzufriedenheit zu verbessern. Die derzeitige Infrastruktur, bestehend aus Kassensystemen, Lagerverwaltung und Kundendatenbanken, wird hauptsächlich zur Datenspeicherung genutzt, jedoch nicht für fortschrittliche Analysen oder Echtzeit-Entscheidungen.

Das Ziel des Projekts ist der Aufbau einer Dateninfrastruktur, die in der Lage ist, kontinuierlich große Mengen an Echtzeitdaten zu verarbeiten und automatisierte Prozesse zu ermöglichen. Ein Schwerpunkt liegt auf der automatisierten Bestandsverwaltung, dynamischen Preisgestaltung und personalisierten Angeboten. Automatisierte Nachbestellungen sollen auf Grundlage von Verkaufsdaten, Lagerbeständen und saisonalen Schwankungen ausgelöst werden.

## 2 Strategie

Um diese Ziele zu erreichen, verfolgt die Frischmarkt AG eine mehrstufige Strategie zur Entwicklung einer effizienten Datenverarbeitungsarchitektur.

Ein zentrales Prinzip der Strategie ist der Aufbau einer modularen und skalierbaren Architektur auf Basis einer Microservice-Architektur. Kubernetes und Helm werden als Werkzeuge für Infrastructure as Code (IaC) verwendet, um die Skalierbarkeit und Wartbarkeit des Systems zu gewährleisten. Kubernetes orchestriert die Microservices, um ihre Skalierung und Fehlertoleranz zu gewährleisten.

Für die Containerisierung wird Podman anstelle der mehr bekannten Docker Container Engine verwendet, da Podman keinerlei proprietäre Einschränkungen wie Docker hat und außerdem einige Vorteile gegenüber Docker bietet, wie z. B. Rootless Containers, welche die Sicherheit erhöhen. Podman kann problemlos im Zusammenhang mit Kubernetes-Umgebungen genutzt werden und unterstützt ebenfalls die wesentlichen Features, die Docker bietet.

Die Architektur teilt das System in unabhängige Komponenten auf, z.B. Datenaufnahme und -verarbeitung, die von Apache Kafka als Nachrichten-Streaming-Plattform übernommen wird. Diese Modularität erleichtert die Skalierbarkeit des Systems, da einzelne Services unabhängig skaliert oder aktualisiert werden können.

Ein zentraler Aspekt des Systems ist die automatisierte Bestands- und Verkaufserfassung. Verkaufs- und Lagerbestandsdaten werden kontinuierlich über Apache Kafka gesammelt und in Echtzeit verarbeitet. Sobald festgelegte Bestandsgrenzen erreicht sind, sollen z. B. automatische Nachbestellungen ausgelöst.

Während der Entwicklungsphase werden Fakes verwendet, um Datenströme zu simulieren und die automatisierte Bestandsverwaltung zu testen, bevor das System mit realen Daten verknüpft wird. Die Einführung dieser Verwaltung würde nach Abschluss schrittweise in Pilotfilialen erfolgen, um die Effizienz und Anpassungsfähigkeit an regionale und saisonale Unterschiede zu testen. Kubernetes ermöglicht die flexible Skalierung der Infrastruktur, um die Datenanforderungen zu bewältigen.

Die datengestützte Entscheidungsfindung wird durch Echtzeit-Datenanalysen ermöglicht, die es dem Management erlauben, auf aktuelle Entwicklungen in den Filialen zu reagieren. Mithilfe von Machine Learning-Algorithmen werden Verkaufsprognosen erstellt, die dynamische Preisanpassungen und personalisierte Angebote

unterstützen. Diese Algorithmen werden kontinuierlich mit aktuellen Verkaufs- und Lagerdaten sowie weiteren relevanten Daten trainiert, die über Kafka in das System eingespeist werden. In der Entwicklungsphase wird dies mit simulierten Daten getestet.

Obwohl Machine Learning eine zentrale Rolle in der zukünftigen Entwicklung spielt, liegt der Fokus dieses Projekts auf der Implementierung der zugrunde liegenden Dateninfrastruktur. Die konkrete Implementierung der Machine Learning Algorithmen wird nicht abgehandelt.

Die nahtlose Integration bestehender Systeme ist ein weiterer strategischer Fokus. Die Kassensysteme und CRM-Systeme der Frischmarkt AG sollen in die neue Dateninfrastruktur eingebunden werden, wobei Apache Kafka als zentrale Plattform für die Datenaufnahme fungiert.

Da der Zugang zu produktiven Datenquellen in der Entwicklungsphase noch nicht gegeben ist, werden wie bereits erläutert Fakes verwendet, um Kassendaten, Lagerbestände und andere Daten zu simulieren. Diese Fakes publizieren Daten an Kafka Topics, um den realen Betrieb zu simulieren und die Datenpipeline unter realistischen Bedingungen zu testen.

Die Ziel-Architektur ist in nachfolgender Abbildung 0.1 zu sehen.

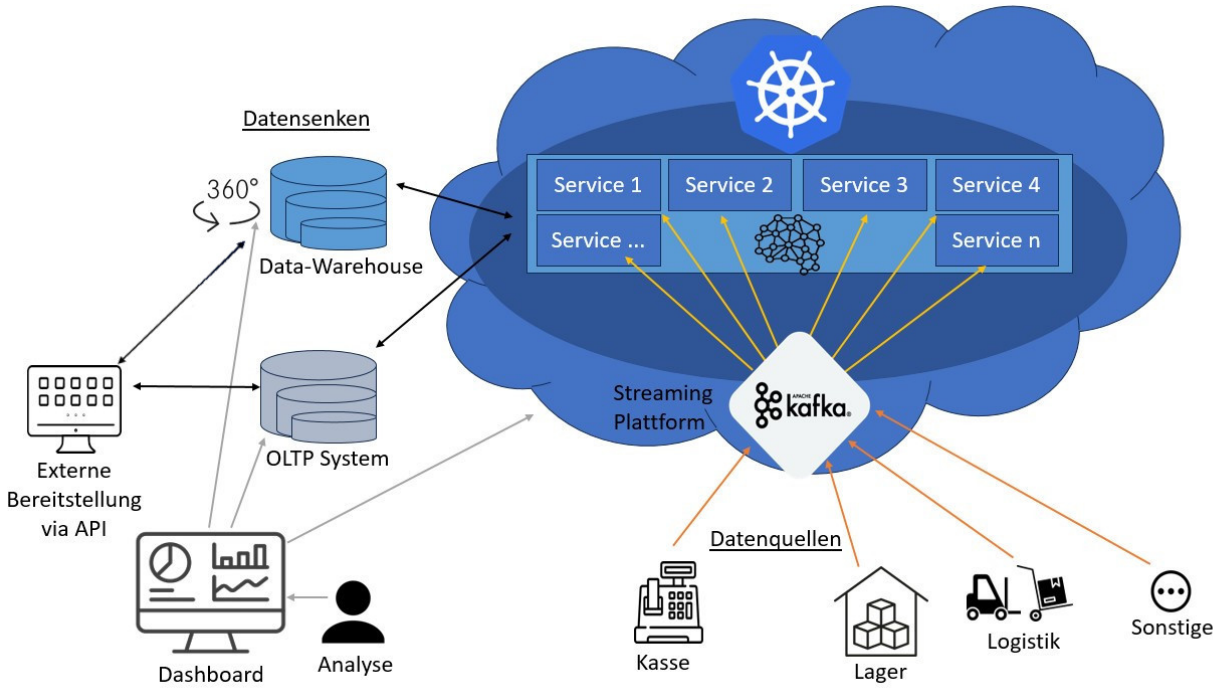


Abbildung 0.1: Architekturübersicht

Datensicherheit und Datenschutz sind ebenfalls zentrale Bestandteile. Erfasste Kundendaten werden gemäß der Datenschutz-Grundverordnung (DSGVO) verarbeitet. Die Speicherung und Verarbeitung der Daten erfolgt verschlüsselt, und es werden strenge Zugangskontrollen implementiert.

Um die Einhaltung der Datenschutzanforderungen sicherzustellen, werden Sicherheitsmechanismen wie verschlüsselte Datenübertragungen und in Zukunft regelmäßige Sicherheitsaudits in die Infrastruktur integriert. Ein Data Governance-Framework regelt den Umgang mit den Daten und gewährleistet Transparenz.

Die Zusammenarbeit zwischen den internen IT-Teams der Frischmarkt AG und externen Experten ist entscheidend für den Erfolg des Projekts. Git in Verbindung mit GitHub wird als Plattform für die Versionskontrolle genutzt, um eine klare Rückverfolgbarkeit der Änderungen und eine kollaborative Entwicklung in verteilten Teams zu ermöglichen.

Die Strategie zur Entwicklung der Dateninfrastruktur für die Frischmarkt AG basiert auf einer modularen, skalierbaren Architektur, die Kubernetes, Helm und Podman verwendet, um die Skalierbarkeit, Wartbarkeit und Verfügbarkeit des Systems sicherzustellen. Apache Kafka wird als zentrale Plattform für die Datenaufnahme und -verarbeitung eingesetzt. Simulierte Datenquellen werden genutzt, um die Datenpipeline während der Entwicklungsphase zu testen. Datensicherheit und Datenschutz haben höchste Priorität, um den gesetzlichen Anforderungen gerecht zu werden.