Bachelorprojekt: BI – Live Data Connection

Entwurf eines SAP Live-Monitoring-Systems

Hochschule Bremerhaven

Projektleiter: Alfred Schmidt - SS2022 / WS2023 - Hochschule Bremerhaven

Unser Projekt

In unserem Bachelorprojekt geht es im Wesentlichen um den Entwurf und die Umsetzung einer docker-basierten Live-Monitoring-Anwendung für Sales-Daten im Kontext des Themas Business Intelligence.

Dabei verfolgten wir zum einen das Ziel, über einen Java Stack und mit ABAP eine Echtzeit-Datenänderungsaufzeichnung (Live-Data-Connection) mit Anbindung an ein SAP/ERP-Systems zu entwerfen. Über einen Java Connector entwickelten wir eine Verbindung zwischen SAP und dem Java-Client auf. Ein zweites Ziel war es, über ein CSV-Connector eine Verbindung zwischen FTP-Server und CSV-Daten herzustellen, damit eine Echtzeit-Datenänderungsaufzeichnung auch hier ermöglicht wird.
Die extrahierten Daten der verschiedenen Quellen sollen dann in einer eigenen Datenbank im Backend konsolidiert und entsprechend transformiert werden.

Die Daten aus dem Backend müssen wiederum über geeignete APIs bedarfsgerecht in unsere Frontend-Anwendung geladen werden, um auch den ETL-Prozess im Business Intelligence Kontext gänzlich abzubilden.

Zuletzt werden in unserer Frontend-Anwendung (unsere Visitenkarte) verschiedene dynamische Sales-Grafiken aufgezeigt. Es wurden über die Grundlagen hinaus wertvolle Erfahrungen gesammelt, wie es ist, sich in einem Team zu integrieren, Verantwortung zu übernehmen und schwierige Situationen als Team zu lösen.

Projekt-Workflow

Unser Team besteht aus 21 Personen und ist deshalb verhältnismäßig deutlich größer als gewohnt. Daher setzten wir uns von Beginn an das Ziel, eine Projektstruktur herzustellen.

Unser Projekt lässt sich in drei Bereiche aufteilen, die wie ein Zahnrad ineinandergreifen, aber zunächst nicht zwangsweise parallel zu entwickeln waren: Frontend - Backend & Live-Daten-Austausch.

Wir entschieden uns dafür, Projektmitglieder unter Berücksichtigung der Kernkompetenzen auf die drei Teams zu verteilen, wodurch angenehme Arbeitsgruppen entstanden.
Das Projekt haben wir in 8 Meilensteine unterteilt, um das zielorientierte Arbeiten mit messbaren Ergebnissen nicht aus den Augen zu verlieren. Zu diesen Meetings haben wir teamübergreifend Ergebnisse bewertet und Kritik und Feedback ausgetauscht.

In Weeklys wurden ToDos herausgearbeitet und Tickets erstellt, geschätzt und auf die Mitglieder verteilt. In einem weiteren wöchentlichen Meeting vor Ort an der Hochschule wurden von allen Teams Fortschritte und Ziele festgehalten.

Business Intelligence

Business Intelligence (BI) wird als Hilfsmittel eingesetzt, indem unternehmensbezogene Prozesse systematisch analysiert werden können. Mit BI können Ergebnisse, Umsätze etc, analytisch durchleuchtet werden. In unserem Beispiel können wir mit Hilfe von BI die vorhandenen Sales Daten analysieren und Erkenntnisse gewinnen. (vgl. Skyrius, 2021)

Technisch gesehen wird zunächst ein ETL-Prozess implementiert. Der ETL-Prozess besteht aus den drei Phasen Extraktion, Transformation und Laden. Der ETL-Prozess hat in der BI das Ziel, Daten aus unterschiedlichen Quellen und Informationssystemen zu sammeln, zu bearbeiten und zusammenzuführen, damit sie von den Benutzern ausgewertet werden können.

Um mit BI Ergebnisse zu erfassen, arbeiten wir mit Kennzahlen, also mit messbaren Werten wie zum Beispiel dem Umsatz oder der Retourenanzahl.

Das Live-Monitoring

Im Hintergrund für das Live-Monitoring können zwei verschiedene Prozesse bedient werden. Der SAP-Prozess und der CSV-Prozess.

Beim SAP-Prozess nutzen wir die Anbindung eines Java-Clients an ein SAP-Server mit ABAP, die wir mit Hilfe eines Java-Connectors (JCO) herstellen können. Der JCO ist unsere Middleware-Komponente, die die Kommunikation zwischen Java-Komponenten und Java-Anwendungsservern mit dem SAP Server über ABAP ermöglicht.

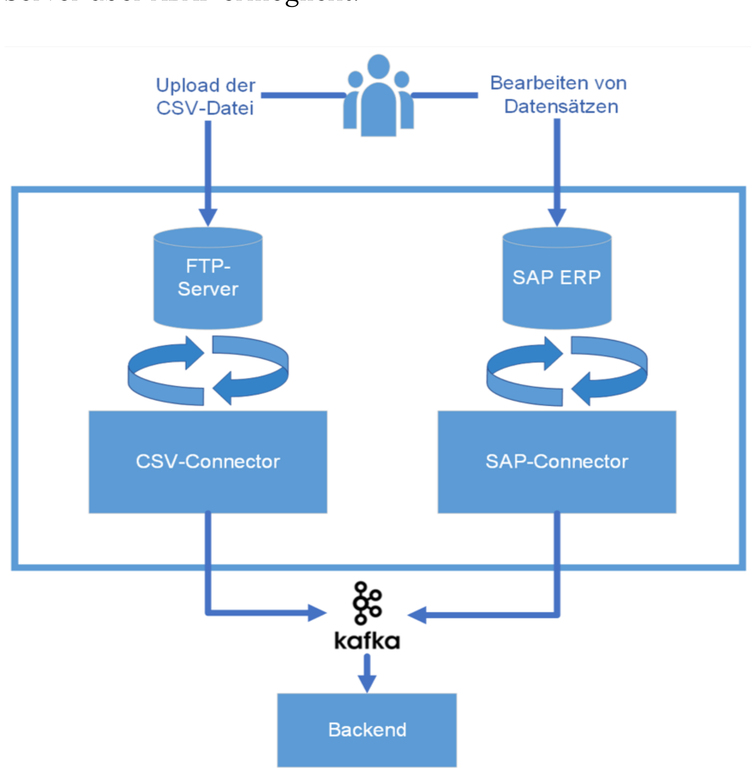

Abbildung 1: Architektur der genutzten Konnektoren

Funktionsweise der Konnektoren

Nachdem die Anbindung über den JCO an den ABAP-Server hergestellt wurde, ist es möglich, Datenveränderungen auf dem ABAP-Server festzustellen. Wenn über die Oberfläche des ABAP-Servers direkt Daten verändert werden, kann die veränderte Tabelle mit Hilfe eines Outbound-Calls übermittelt werden.
Über Java werden für jede einzelne Zeile Datentransferobjekte erstellt, die dann an Kafka weitergeleitet werden.

Der SAP Java-Connector (JCO) ist eine Middleware-Komponente, die die Kommunikation zwischen Java-Komponenten und Java-Anwendungsservern mit dem SAP Server ABAP ermöglicht.
Durch einen Inbound-Call schickt der Java Client an den ABAP Server einen Request heraus. Die Antwort ist ein Outbound-Call des ABAP Servers an den Java Client.
So können Server und Client über Desktop- oder Serveranwendung miteinander kommunizieren.

Der CSV-Prozess erfolgt durch einen manuellen Import einer Datei auf den FTP-Server. Eine CSV-Datei wird auf den Server hochgeladen und im Hintergrund erfolgt eine regelmäßige Abfrage, ob Veränderungen zu der Altdatei festgestellt werden können. Sobald eine Veränderung erfolgt, werden auch hier mit Hilfe von Java Datentransferobjekte erstellt, die dann weiter an Kafka gesendet werden. Die Datei wird dann dann automatisch aus dem Pfad des Ordners verschoben.

Was passiert im Hintergrund?

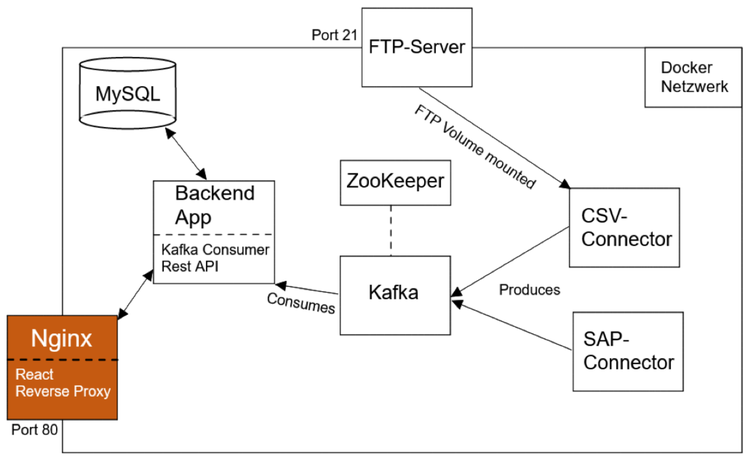

Abbildung 2: Gesamtarchitektur zum Projektaufbau

Um unser Ziel, Sales-Daten in Echtzeit zu visualisieren, müssen Frontend, Backend das Live-Data-Monitoring bei Datenübertragungen etc. das gleiche Format nutzen. Daher war es eine der Hauptaufgaben des Backends, zunächst einen ETL-Prozess aufzubauen, indem die Daten erfasst (Extraktion), im weiteren Verlauf transformiert und nach der Transformation also APIs bereitgestellt werden (Ladeprozess).

Angewendete Technologien

Also Technologien kamen hauptsächlich ein gesammelter Java Stack bestehend aus dem Framework Spring Boot, Maven und Java selbst vor, um unter anderem die Anbindung der Konnektoren herzustellen.

Zudem nutzten wir neben Docker auch Kafka als Lösungstool, um uns vor den Problemen bei der langsamen Verarbeitung von großen Datensätzen zu schützen. Kafka ist ein sehr beliebtes Tool im Big-Data Umfeld und nutzt eigene Cluster. Kafka stellt Schnittstellen bereit, um große Datenströme in oder aus Drittsystemen zu importieren oder exportieren. (vgl. Luber, 2018)

Die Persistenz von Kafka ist ebenfalls gut. Bei unserem docker-basierten Setup wurden 130.000 Datentransferobjekte in knapp unter 4 Sekunden an Kafka gesendet.
Für das Frontend nutzten wir die Technologien NGINX, TypeScript, vite js und React.

Unser Projektfazit

Das Projekt zur Entwicklung einer Live-Monitoring-Anwendung im SAP Bereich war für uns ein Erfolg. Wir haben uns in einem Zeitraum von 10 Monaten erfolgreich mit mehreren Technologien im SAP und Java Bereich auseinandersetzen können und viel Erkenntnisse gewinnen können.
Trotz einiger Herausforderungen und dem Umstand, dass vielleicht nicht immer alle Mitglieder gleichermaßen motiviert waren, haben wir es geschafft, gemeinsam ein erfolgreiches Projekt abzuschließen.

Das Projektteam bestand aus 21 Personen, die alle ihre Stärken und Fähigkeiten eingebracht haben. Insgesamt war es eine großartige Erfahrung und ein sehr praxisnahes Projekt, das aus unserer Sicht ein wertvoller Beitrag zur Informatik an unserer Hochschule war.

Frontend

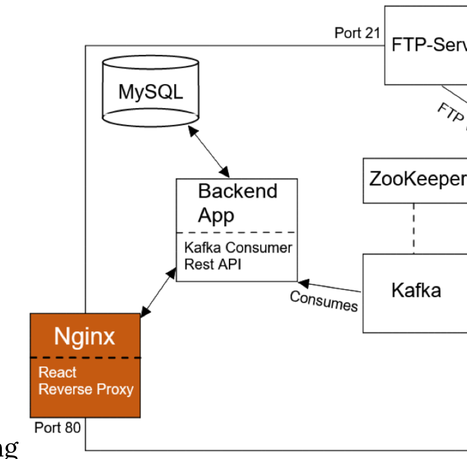

Bild unserer Frontendanwendung

Abbildung 3: Vorschaubild unserer Frontendanwendung

Literatur

Luber, S. (2018). *Apache Kafka*. Verfügbar 20. Januar 2023 unter https://www.bigdata-insider.de/was-ist-apache-kafka-a-763300

Skyrius, R. (2021). *Business Intelligence*. Springer Cham. https://doi.org/10.1007/978-3-030-67032-0