HAW HAMBURG

Bachelorarbeit

Steffen Clemens

Analyse und Visualisierung des Mehrwertes von neuen Features in Webanwendungen

Fakultät Technik und Informatik
Department Informatik

Faculty of Computer Science and Engineering
Department Computer Science

Steffen Clemens

Thema der Arbeit

Analyse und Visualisierung des Mehrwertes von neuen Features in Webanwendungen

Stichworte

Feature, Dashboard, Visualisierung, Konfiguration, Data Pipeline, Verteilte Systeme, Webanwendung, E-Commerce

Kurzzusammenfassung

Das Ziel dieser Arbeit ist die Umsetzung eines Systems, welches den Mehrwert von Features bestimmen kann, die in Webanwendungen implementiert wurden. Entscheidend ist, dass Metriken für neue Features mit geringem Zeitaufwand durch Konfiguration hinzugefügt werden können. Die Ergebnisse sollen visuell auf Dashboards dargestellt werden und einen schnellen Eindruck über den Mehrwert der Features vermitteln, aber auch ein Feststellen der Gründe für bestimmte Verläufe oder Ereignisse ermöglichen.

Steffen Clemens

Title of Thesis

Analysis and visualization of the added value of new features in web applications

Keywords

Feature, Dashboard, Visualization, Configuration, Data Pipeline, Distributed Systems, Web Application, E-Commerce

Abstract

The goal of this thesis is to develop a system that can determine the added value of features implemented in web applications. It is crucial that metrics for new features can be added with little time by configuration. The results should be displayed visually on dashboards and give a quick impression of the added value of the features, but also allow to determine the reasons for certain trends or events.

Inhaltsverzeichnis

Abbildungsverzeichnis

Quellcodeverzeichnis

1 Einleitung

1.1 Motivation

Genaue Kennzahlen zu erhalten, welche Mehrwerte bestimmte Features in Softwareprojekten erzielen, hat einige Vorteile. Sie können motivierend auf die Mitarbeiter wirken, aber auch die Kundenzufriedenheit kann besser vorhergesehen und auf Grundlage der gemessenen Erfolge gesteigert werden. Darüber hinaus kann der Kunde durch die Erfahrungen und Erkenntnisse aus dem Dashboard besser bei der Planung neuer Features beraten werden.

1.2 Zielsetzung

Ziel ist es, intuitiv verständliche Dashboards zu erstellen, welche die Feature-Adaption aufzeigen und Risiken erkennen.

Auf den Dashboards soll die Mehrwertbildung durch neue Features sichtbar sein. Ein neues Feature in einem Onlineshop kann beispielsweise eine neue Zahlungsart sein. In diesem Fall vermittelt das Dashboard eindeutige Werte, wie ausgeprägt die neue Zahlungsart genutzt wird. Zum Beispiel kann dies durch die Anzahl oder den prozentualen Anteil von getätigten Bestellungen über diese Zahlungsart gemessen werden. Der Mehrwert des neuen Features kann auch anhand der Umsatzdifferenz festgestellt werden. Jedoch spielen weitere Aspekte, wie zum Beispiel die Saisonalität, eine Rolle und können sich wesentlich auf die Metriken auswirken. Die gemessenen Mehrwerte können zur Mitarbeitermotivation dienen und beispielsweise in Scrum-Retrospektiven vorgestellt werden.

1.3 Gliederung

Die Arbeit ist in sieben Kapitel unterteilt. Nachdem dieses Kapitel das Thema der Arbeit eingeleitet hat, folgt das zweite Kapitel, welches Grundlagen zum Verständnis der Arbeit vermittelt. Das dritte Kapitel enthält die spezifischen Anforderungen an das im Zuge der Arbeit entwickelte System. Im vierten Kapitel wird der Entwurf und die Architektur des Systems beschrieben. Kapitel fünf erläutert die Umsetzung des Systems. Im sechsten Kapitel wird die Umsetzung und der Mehrwert des Systems evaluiert. Kapitel sieben soll abschließend ein Fazit ziehen und Ausblick auf den möglichen weiteren Verlauf des Projekts bieten.

2 Grundlagen

Dieses Kapitel enthält Grundlagen zum Verständnis der Arbeit. Es werden Informationen zu den eingesetzten Konzepten, Technologien und Tools vermittelt.

2.1 Data Pipeline

Zunächst wird beschrieben wie eine *Data Pipeline* aufgebaut wird und was der ETL-Prozess ist. Die Data Pipeline ist ein Kernelement bei der Auswertung der Feature-Metriken.

Eine Data Pipeline ist ein mehrstufiger Prozess der Daten konsumiert und verarbeitet. Die verarbeiteten Daten werden meist in einer aggregierten Form abgespeichert. Diese aggregierten Daten können bei Entscheidungen unterstützen. (Papp, 2019, S. 125)

2.1.1 Zustand der Quelldaten

Data Pipelines lassen sich danach kategorisieren, wie und in welchem Zustand Daten angeliefert werden.

Werden die Daten als Datei angeliefert, können diese nicht direkt verarbeitet werden, da es nicht möglich ist Abfragen auf die Daten auszuführen. Die dateibasierte Speicherung ist einfach umzusetzen, weil sie von keinen weiteren Komponenten abhängig ist und weniger Verarbeitungsschritte für die Speicherung benötigt werden. Die gespeicherten Daten können einfach adaptiert werden, da sie keinem fixen Schema folgen. (Papp, 2019, S. 125)

Die Daten können auch in einer Datenbank gespeichert werden. Sie können dann leicht extrahiert, transformiert und geladen werden. Dazu müssen die Daten ein vorher bekanntes Schema aufweisen oder in ein entsprechendes gebracht werden. (Papp, 2019, S. 126)

Wenn es sich als schwierig erweist, die Daten in datenbanktaugliche Strukturen zu konvertieren, bietet sich der Data-Lake-Ansatz an. Näheres zu diesem Ansatz wird in Abschnitt 2.2 beschrieben. Die Daten werden in ihrem ursprünglichen Zustand gespeichert und können anschließend in verschiedene Schemata überführt werden. Dadurch wird eine hohe Flexibilität der Daten in der Data Pipeline erreicht. (Papp, 2019, S. 126)

Eine hohe Flexibilität bei der Zusammenstellung der Komponenten der Data Pipeline bietet der Serverless-Ansatz. Teile der Data Pipeline werden in einzelnen Containern gekapselt, die dynamisch hinzugefügt, entfernt und skaliert werden können. Dieser Ansatz ist momentan vergleichsweise noch sehr teuer. (Papp, 2019, S. 126)

Der Data-Lake-Ansatz ist derzeit State of the Art unter den Data-Pipeline-Konzepten (Papp, 2019, S. 126).

2.1.2 Aufbau einer Data Pipeline

Abbildung 2.1 zeigt den grundsätzlichen Aufbau, der bei allen Pipelines konzeptionell ähnlich ist (Papp, 2019, S. 128).

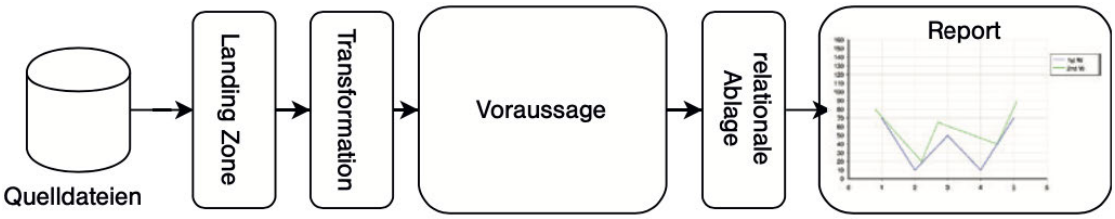


Abbildung 2.1: Data Pipeline Konzept (nach Bild 4.1 aus Papp, 2019, S. 127)

Der Prozess des Kopierens der Daten von Quellsystemen auf ein Zielsystem, welches die Daten anders darstellt als in der Quelle, wird als ETL-Prozess bezeichnet. ETL steht für extract, transform und load. (Denney u. a., 2016, S. 272)

Anfangs werden die Daten von den Quellsystemen geladen und zunächst in einer Staging Area, auch Landing Zone genannt, abgelegt (Oracle Corp., 2002, S. 272). Dies entspricht dem extract-Schritt des ETL-Prozesses. Eine Staging Area ist ein Zwischenspeicher, der meist nach Abschluss des Prozesses gelöscht wird.

Anschließend werden die Daten transformiert, sodass sie in einer Form sind, in der sie in den Zielspeicher geladen werden können. Dies ist der transform-Schritt des ETL-Prozesses. Dieser Schritt umfasst die Bereinigung der Daten, sodass nur die gewünschten

Daten weiterverarbeitet werden. Die Daten werden in diesem Schritt auch in die richtigen Datenformate gebracht. Weitere Funktionen zum Vorbereiten der Daten auf den aktuellen Anwendungsfall können in diesem Schritt durchgeführt werden. Zur Voraussage können diverse Modelle auf den Daten angewandt werden (Papp, 2019, S. 128).

Die Ergebnisse werden abgespeichert, wobei nur die aggregierten Daten gespeichert werden und die Rohdaten im Data Lake verbleiben (Papp, 2019, S. 128). Dies ist der load-Schritt des ETL-Prozesses. Anhand der nun vorliegenden Daten können Reports erzeugt werden, die neue Erkenntnisse liefern.

2.1.3 Problemstellungen und Wertgenerierung

Typische Problemstellungen für Data Pipelines sind Optimierungs- und Risikoanalysen. Durch diese können Kosten eingespart und Risiken vermieden werden, was wiederum zu einer Wertgenerierung führt. (Papp, 2019, S. 129)

2.1.4 Data-Science-Prozess

Der Data-Science-Prozess soll Informationen aus der realen Welt aufbereiten, sodass daraus Erkenntnisse gewonnen werden können, um die Realität zu optimieren.

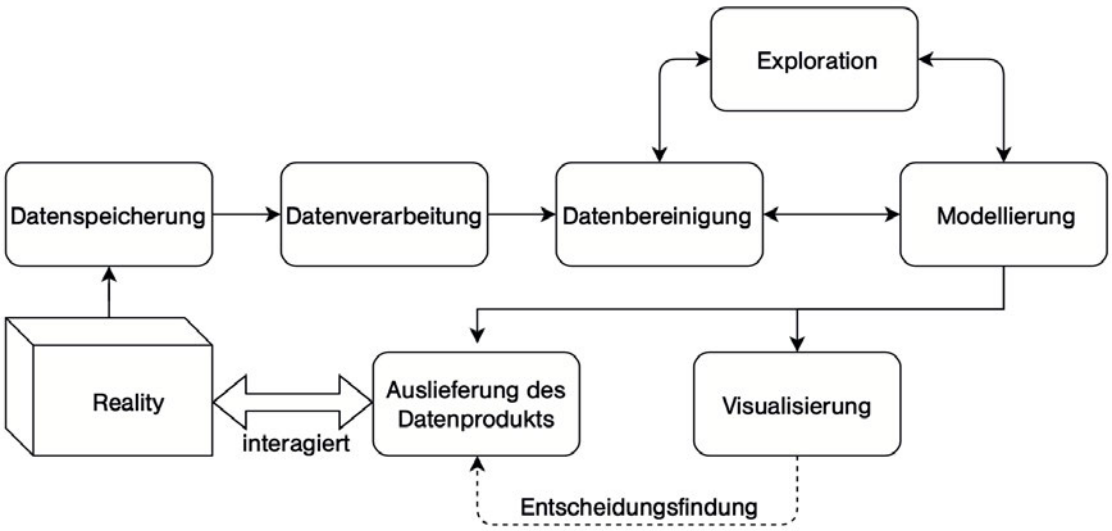


Abbildung 2.2: Data-Science-Prozess (nach Bild 4.3 aus Papp, 2019, S. 130)

Abbildung 2.2 zeigt den Data-Science-Prozess. Die Daten werden aus Systemen der realen Welt erhoben und gespeichert. Anschließend werden sie verarbeitet. Daraufhin werden Modelle gebaut und Visualisierungen erzeugt, auf dessen Grundlage wiederum Entscheidungen getroffen werden können. In der Abbildung wird deutlich, dass das Ergebnis des Prozesses wiederum mit der Realität interagiert, indem die gewonnenen Informationen oder getroffenen Entscheidungen Einfluss darauf nehmen.

2.2 Data Lake

Das Konzept des *Data Lake* wird beschrieben, da für die Auswertung von Features eine Vielzahl von Daten abgerufen werden muss. Diese Daten sollen nicht doppelt abgerufen werden müssen, selbst wenn zu späteren Zeitpunkten Werte aus den ursprünglichen Rohdaten benötigt werden, welche zum jetzigen Zeitpunkt nicht relevant erscheinen. Es ist daher sinnvoll, die abgerufenen Daten in ihrer Rohform vorzuhalten.

Ein Data Lake ist ein Speicher, der alle Unternehmensdaten als Rohkopien von unterschiedlichen Quellsystemen sowie transformierte Daten aufnimmt. Diese dienen für Aufgaben wie Reporting, Visualisierung, fortschrittliche Analyse und Machine Learning. (Strengholt, 2020, S. 301)

Im Folgenden werden typische Anforderungen beschrieben, bei denen ein Data Lake zum Einsatz kommen kann.

In vielen Fällen müssen Unternehmen mit strukturierten und semi-strukturierten Daten umgehen, die sich nur schwer in einer gemeinsamen Datenbank vereinen lassen. Quelle der Daten ist typischerweise eine Anwendung oder ein System, häufiger werden auch Daten aus verschiedenen Systemen zusammengetragen. Die Daten sind in der Regel subtransaktional oder nicht-transaktional. Das bedeutet, dass Transaktionen geschachtelt sein können, sodass innerhalb einer Transaktion weitere gestartet werden oder Daten ohne Zugehörigkeit zu einer Transaktion verändert werden. Es handelt sich dabei beispielsweise um Informationen, wie eine Person auf eine Website gekommen ist und wo sie den Bestellen-Button geklickt hat. (Louwers, 2012)

Es gibt einige bekannte Fragen, die durch Analyse der Daten beantwortet werden sollen. Jedoch gibt es viele noch unbekannte Fragen, die erst in der Zukunft aufkommen, sowie verschiedene Anwender*innen, die Antworten aus den Daten erhalten möchten. In vielen Fällen ist die Menge der Daten oder das tägliche Aufkommen an Daten so groß, dass es

nicht technisch oder ökonomisch von einem relationalen Datenbankmanagementsystem aufgenommen werden kann. (Dixon, 2010)

Verfolgt man den Ansatz, nur die aktuell interessantesten Attribute zu untersuchen und aggregiert abzuspeichern, verliert man die Fähigkeit zukünftige Fragen anhand der Daten zu klären und tiefere Einblicke in die Daten zu erhalten (Dixon, 2010).

Dieses Problem löst das Data Lake Konzept, indem die Rohdaten, nachdem sie verarbeitet und die Ergebnisse abgespeichert wurden, weiterhin im Data Lake verbleiben. Die aggregierten Daten sind mit den Rohdaten verknüpft, sodass sie nachgeladen werden können, wenn Änderungen auftreten. (Papp, 2019, S. 128)

Wenn neue Daten im Data Lake abgespeichert werden, muss dies effizient passieren. In diesem Schritt sollten keine oder kaum Transformationen wie Validierung oder Formatierung der Daten ausgeführt werden und alle Daten in ihrer ursprünglichen Form abgelegt werden (Papp, 2019, S. 130). Es sollten nur die notwendigen Daten abgerufen werden, um den Datenbestand im Data Lake aktuell zu halten. Wie erkannt werden kann, welche Daten sich verändert haben, wird in *Abschnitt 2.3* erläutert.

2.3 Change Data Capture

Beim Abrufen der Daten ist es wichtig, dass es Mechanismen gibt, die verhindern, dass operative Systeme durch den Datenexport zu sehr belastet werden (Papp, 2019, S. 130). Die Last auf die Systeme beim Abrufen von Daten kann durch *Change Data Capture* deutlich verringert werden. Da sich die Daten auf den operativen Systemen stetig verändern, ist es notwendig, diese Veränderungen festzustellen. So ist es möglich nur die veränderten und neuen Daten abzurufen.

Change Data Capture (CDC) ist ein Ansatz, um nur die veränderten Daten von Systemen abzurufen (Tank u. a., 2010, S. 367). Hierbei werden häufig Transaction Logs verwendet, um den Abruf der Daten zu steuern (Papp, 2019, S. 130). Diese beschreiben die Veränderungen, welche in der Datenbank vorgenommen wurden. Sie stehen jedoch nicht immer zur Verfügung oder der Zugang zu ihnen ist beschränkt. Häufig enthalten Datensätze aber einen Zeitstempel, der den Zeitpunkt der letzten Veränderung enthält. Merkt man sich den Zeitstempel vom letzten Abruf, kann man danach filtern, um nur Daten zu erhalten, welche später aktualisiert und somit noch nicht abgerufen wurden.

2.4 Elasticsearch

Für die Speicherung der transformierten Daten kann *Elasticsearch* eingesetzt werden. Zum besseren Verständnis wird die Funktionsweise von *Elasticsearch* erklärt.

Elasticsearch ist ein quelloffenes Suchserver-Projekt, welches im Februar 2010 veröffentlicht wurde. Aufgrund seiner verteilten Struktur und Echtzeitfähigkeit, wird es von vielen als Dokumentendatenbank verwendet. (Kuć, 2014, S. 12)

2.4.1 Indizes

Ein Index in Elasticsearch ist vergleichbar mit einer Tabelle in einer relationalen Datenbank. Die Werte in einem Index sind jedoch für eine schnelle und effiziente Volltextsuche vorbereitet. Ein Index kann außerdem Dokumente verschiedener Typen enthalten, wie beispielsweise Bestellungen und Kunden. (Kuć, 2014, S. 12)

Damit die Suche schnell und effizient funktionieren kann, wird ein invertierter Index aufgebaut, ähnlich dem Stichwortverzeichnis am Ende eines Buches. Anstatt den ganzen Text des Buches nach dem Suchbegriff zu durchsuchen, wird der Index durchsucht, dessen Einträge auf die gewünschten Ergebnisse verweisen. Um den invertierten Index aufzubauen, verwendet Elasticsearch die Suchmaschinen-Bibliothek *Apache Lucene*. (Divya und Goyal, 2015, S. 173)

2.4.2 Dokumente

Ein Dokument in Elasticsearch ist zu verstehen wie eine Tabellenzeile einer relationalen Datenbank. Dokumente enthalten Felder, vergleichbar mit Tabellenspalten einer relationalen Datenbank. Felder können jedoch auch mehrfach in einem Dokument vorkommen, ein solches Feld ist dann mehrwertig. Jedes Feld hat einen Typen, beispielsweise enthält es Text, einen Zahlenwert oder ein Datum. Es sind aber auch komplexe Feldtypen möglich, wie Listen oder Unterdokumente. Anhand der Feldtypen kann Elasticsearch feststellen, wie Operationen, wie das Vergleichen der Daten, durchgeführt werden können. (Kuć, 2014, S. 12f.)

2.4.3 Mapping

Elasticsearch indiziert gespeicherte Dokumente entweder durch ein dynamisches Mapping oder durch ein Mapping, welches man zuvor festgelegt hat. Mit letzterem lassen sich meist die besten Ergebnisse erzielen. (Divya und Goyal, 2015, S. 172) Ein Mapping kann beispielsweise festlegen, nach welchem Format ein Datumswert geparsed wird, welche Felder Schlüsselwörter enthalten oder in welches Zahlenformat Werte übertragen werden sollen.

2.4.4 REST-API

Die Anfragen an Elasticsearch laufen über eine REST-API. Diese unterstützt PUT, POST, GET und DELETE Methoden, um Operationen mit den Daten auszuführen. Beim Verändern eines Dokuments mit der Methode PUT wird eine Versionsnummer hochgezählt, sodass geprüft werden kann, ob es sich um die erwartete Version eines Dokuments handelt oder es in der Zwischenzeit verändert wurde. Beim Hinzufügen eines Dokuments mit der Methode POST kann eine ID generiert werden, sofern keine vorgegeben wurde. Mit der Methode GET können angelegte Dokumente wie in einer Schlüssel-Werte-Datenbank nach dem Schema index/typ/id abgefragt werden. (Divya und Goyal, 2015, S. 172)

Beispielsweise kann ein Index mit dem Bezeichner „orders“ für Bestellungen angelegt werden. In diesem Index können Dokumente von verschiedenen Typen angelegt werden, zum Beispiel „catalogorder“ und „order“. Darunter werden Daten von Katalogbestellungen, bei denen ein Katalog in Papierform versendet wird und die kostenlos sind, oder normale Bestellungen, bei denen die eigentlichen Produkte verkauft werden, als Dokument abgelegt. Jedes Dokument wird mit einer eindeutigen ID abgespeichert, unter der es über die REST-API erreichbar ist.

2.4.5 Suche

Für die Suche steht eine gesonderte Ressource zur Verfügung. Das Indizieren wird aus Gründen der Effizienz in der Massenverarbeitung durchgeführt. Im Gegensatz zum Abfragen eines Dokuments anhand seiner ID, sind Veränderungen daher nicht sofort in den Suchergebnissen sichtbar. Die Suche kann auf drei Ebenen durchgeführt werden, der Server-, Index- oder Typ-Ebene. Die Such-API ist erreichbar unter der Ressource

_search, welche unter den drei Ebenen verfügbar ist. Für einfache Suchen kann eine Lucene Anfrage mit dem Query-Parameter q in der URL übergeben werden. Komplexere Anfragen können anhand einer domänenspezifischen Sprache als JSON-Body ausgeführt werden. (Divya und Goyal, 2015, S. 172)

2.5 Grafana

Für die Visualisierung von Kennzahlen bietet *Grafana* umfangreiche Möglichkeiten zur grafischen Darstellung der Feature-Informationen.

Grafana ist eine Open-Source-Software zur Visualisierung und Analyse von Daten (Grafana Labs, 2021e). Die Daten können aus verschiedenen Datenquellen abgefragt werden, unter anderem aus *Elasticsearch*. Visualisiert werden die Informationen auf dafür angelegten Dashboards.

Ein Dashboard ist eine visuelle Darstellung der wichtigsten Informationen, die zum Erreichen eines oder mehrerer Ziele benötigt werden. Die Informationen werden auf einem einzigen Bildschirm gesammelt und angeordnet, sodass sie auf einen Blick überwacht werden können. (Few, 2006, S. 26)

2.5.1 Datenquellen

Grafana kann Daten aus vielen verschiedenen Datenquellen einbinden. Wichtig dabei ist, dass die Daten einen Zeitbezug haben, sodass sie in Diagrammen zeitlich eingeordnet werden können. Dazu wird für die meisten Datenquellen beim Hinzufügen das Feld festgelegt, welches die Zeitinformation enthält.

Für jede Datenquelle gibt es einen spezifischen Abfrage-Editor. Über diesen können die spezifischen Fähigkeiten der jeweiligen Art der Datenquelle genutzt werden, um Daten abzufragen. Daten aus mehreren Datenquellen können auf einem Dashboard kombiniert werden. (Grafana Labs, 2021d)

2.5.2 Panels

Panels sind die grundlegenden Visualisierungsbausteine in Grafana. Ein Panel hat einen Abfrage-Editor. Dieser ermöglicht die Konfiguration, wie die Visualisierung aus den Daten erzeugt werden soll, welche auf dem Panel angezeigt wird. Eine Vielzahl von Styling- und Formatierungsoptionen ermöglichen dann die zielgenaue Darstellung der gesammelten Informationen. (Grafana Labs, 2021g)

Ein Abfrage-Editor ist ein Formular, das beim Schreiben der Abfrage unterstützt. Die Abfrage wird in der Abfragesprache geschrieben, die von der Datenquelle verwendet wird. Darüber hinaus gibt es weitere Abfrage-Optionen, die beispielsweise festlegen, wie viele Datenpunkte gesammelt werden sollen. (Grafana Labs, 2021h)

2.5.3 Dashboards

Ein Dashboard in Grafana ist ein Satz von *Panels*, die in Reihen angeordnet sind (Grafana Labs, 2021c). Die Panels können auf dem Dashboard verschoben und in ihrer Größe verändert werden (Grafana Labs, 2021g). Abbildung 2.3 zeigt ein beispielhaftes Grafana-Dashboard. Mit der 4 wird ein Panel markiert.

Abbildung 2.3: Beispielhaftes Grafana-Dashboard (aus Grafana Labs, 2021c, S. 18)

Das Dashboard bietet mehrere Optionen zur Wahl des anzuzeigenden Zeitbereichs. Der gewählte Zeitbereich wird auf alle Panels auf dem Dashboard angewendet. Über ein Dropdown-Feld kann ein relativer Zeitbereich festgelegt werden (siehe 2 in Abbildung 2.3),

sodass zum Beispiel stets Informationen über die letzten 24 Stunden angezeigt werden. Auch absolute Zeitbereiche können festgelegt werden. Dazu kann auf einem beliebigen Diagramm, welches eine Zeitachse besitzt, durch Ziehen mit dem Cursor in einen Bereich hineingezoomt und so der Bereich für das gesamte Dashboard aktualisiert werden. Für das Herauszoomen und Verschieben des Zeitbereichs gibt es Schaltflächen (siehe 1 in Abbildung 2.3). (Grafana Labs, 2021c)

Die Aktualisierung der Informationen auf dem Dashboard kann manuell über eine Schaltfläche durchgeführt werden (siehe 3 in Abbildung 2.3) oder es kann ein Intervall gewählt werden, in welchem aktualisiert wird. In Kombination mit der Wahl eines relativen Zeitbereichs können auf diese Weise stets aktuelle Daten auf dem Dashboard angezeigt werden. (Grafana Labs, 2021c)

Für ein Dashboard können Annotationen festgelegt werden, die in Zeitverlaufsdiagrammen angezeigt werden. Dies ist eine Möglichkeit um Ereignisse zu markieren und Relationen zwischen Ereignissen und bestimmten Kennzahlverläufen herzustellen. Bewegt man den Cursor über eine Annotation, erhält man nähere Informationen über das Ereignis. (Grafana Labs, 2021b)

2.5.4 Alerts

Alerts ermöglichen es, über Ereignisse und Probleme kurz nach deren Auftreten informiert zu werden. Dadurch können Probleme schnell identifiziert und behoben werden. (Grafana Labs, 2021a)

Durch eine Regel wird festgelegt, unter welcher Bedingung ein Alert ausgelöst werden soll, zum Beispiel wenn ein Grenzwert überschritten wird. Es wird dann über einen festgelegten Kanal eine Benachrichtigung gesendet. Wurden Benachrichtigungsrichtlinien festgelegt, werden Benachrichtigungen gegebenenfalls gruppiert oder nur in einer bestimmten Häufigkeit gesendet. Es können zudem Bedingungen festgelegt werden, unter denen die Benachrichtigungen stumm geschaltet werden. (Grafana Labs, 2021a)

2.5.5 HTTP-API

Der Zugriff auf Grafana ist über eine HTTP-API möglich. Dies ist dieselbe API, welche vom Grafana-eigenen Frontend verwendet wird. Auf diese Weise kann Grafana von an-

deren Programmen gesteuert werden, die beispielsweise Datenquellen aktualisieren oder Dashboards anlegen wollen. (Grafana Labs, 2021f)

2.6 Informationsdesign

Damit aussagekräftige und zielgruppengerechte Visualisierungen erstellt werden können, wird das *Informationsdesign* behandelt. Es wird darauf eingegangen, welche Vorgehensweise es gibt, um die Daten auszuwählen und anhand der richtigen Visualisierungstechnik darzustellen.

Informationsdesign beschäftigt sich mit der Auswahl, Organisation und Präsentation von Informationen, sodass diese effektiv und effizient genutzt werden können. Insbesondere geht es in diesem Abschnitt um das Design von Informationen auf Dashboards. Die Herausforderung bei der Gestaltung von Dashboards besteht darin, eine große Menge nützlicher und meist ungleichartiger Informationen auf kleinem Raum zu platzieren und dennoch Übersichtlichkeit zu bewahren (Few, 2006, S. 81).

Ein Dashboard ist auf das Geschäftsziel der Betrachtenden ausgerichtet und hilft ihnen dieses zu erreichen. Der Fokus auf das Ziel ist wichtig und es sollte vermieden werden viele Daten zu visualisieren, nur um die grafischen Fähigkeiten des Dashboards zu demonstrieren. Stattdessen werden die Daten mithilfe von Visualisierungen zusammengefasst und auf die relevanten Informationen beschränkt. (Janes u. a., 2013, S. 17) Nicht-datenbezogene Bereiche, wie beispielsweise Dekorationen und Hintergründe, die nicht zum besseren Verständnis beitragen, sollte man weitestgehend reduzieren und die wichtigsten datenbezogenen Bereiche hervorheben (Few, 2006, S. 86). Entscheidend für ein gutes Dashboard ist somit die Auswahl der Daten und die Wahl der Visualisierungstechnik (Janes u. a., 2013, S. 17).

2.6.1 Auswahl der Daten

Die Entscheidung, was gemessen werden soll, muss auf dem erwarteten Nutzen der Messung basieren. Um zu bestimmen, welche Fragen beantwortet werden sollen und welche Daten dafür erhoben werden müssen, kann das GQM-Modell angewendet werden. GQM steht für „Goal“, „Question“ und „Measurement“. Das Modell besteht aus einer konzeptionellen, einer operativen und einer quantitativen Ebene. Die konzeptionelle Ebene definiert

das Ziel und somit was untersucht werden soll und warum. Die operative Ebene definiert die Fragestellungen und damit welche Teile des Untersuchungsgegenstandes relevant sind und welche Eigenschaften dieser Teile zur Bewertung verwendet werden. Die quantitative Ebene definiert die Maßnahmen, genauer gesagt welche Daten gesammelt werden müssen, um die Fragen quantitativ zu beantworten. (Janes u. a., 2013, S. 18)

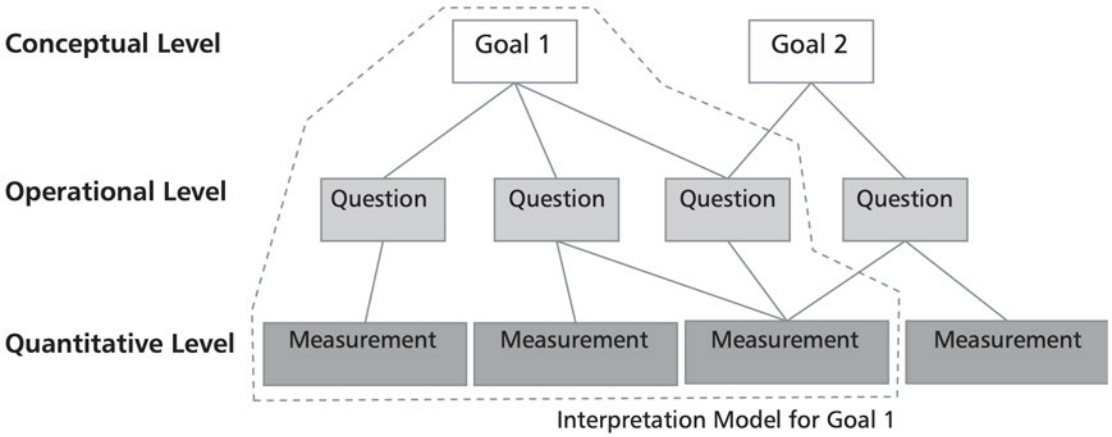


Abbildung 2.4: GQM-Modell (nach Bild 3 aus Janes u. a., 2013, S. 18)

Abbildung 2.4 zeigt die drei Ebenen des GQM-Modells als Hierarchie von Zielen, Fragen und Messungen. Die Ebenen werden durch ein Interpretationsmodell ergänzt, welches die Interpretation der gesammelten Daten definiert. (Janes u. a., 2013, S. 18)

Das GQM-Modell lässt sich zum GQM+Strategies-Ansatz erweitern. Messziele können auf verschiedenen Ebenen der Organisation auftreten und lassen sich in einer Zielhierarchie einordnen. Ein Hauptziel einer Organisation kann die Steigerung des Umsatzes sein. Daraus lassen sich neue Ziele formulieren, die zur Erfüllung dieses Ziels führen, wie zum Beispiel die Steigerung des Ansehens des Unternehmens. Dieses ließe sich möglicherweise durch bessere Angepasstheit der implementierten Features erreichen. Dies lässt sich fortsetzen. Durch die Definition einer solchen Zielhierarchie werden abstrakte Ziele anhand von Mittel-Zweck-Beziehungen detaillierter beschrieben. So können Geschäftsziele gemessen werden, sowohl auf tieferen Ebenen anhand detaillierterer Ziele, als auch auf strategischer Ebene anhand abstrakterer in Beziehung stehender Ziele. (Janes u. a., 2013, S. 19f.)

2.6.2 Wahl der Visualisierungstechnik

Die Gestaltung eines Dashboards hängt stark von den Anforderungen ab, die es erfüllen muss. Grundlegend gibt es zwei Szenarien, wie ein*e Benutzer*in bestimmte Informationen erhält. (Janes u. a., 2013, S. 20)

Im Pull-Szenario betrachtet die oder der Benutzer*in das Dashboard auf der Suche nach einer bestimmten Information und bedient es gegebenenfalls, um die Information zu erhalten. In diesem Szenario ist die wahrgenommene Nützlichkeit und Benutzerfreundlichkeit besonders wichtig. Das Dashboard sollte so gestaltet sein, dass es der benutzenden Person hilft, zu verstehen, wie sie die Daten interpretieren kann. Es sollte für diese einen minimalen Aufwand erfordern, um die vermittelte Botschaft zu erhalten und die Möglichkeit bieten, den Detaillierungsgrad der Informationen zu wählen. (Janes u. a., 2013, S. 20)

Ein Dashboard für das Push-Szenario muss so gestaltet sein, dass wichtige Informationen die Aufmerksamkeit der Benutzerin oder des Benutzers erregen. Das Dashboard sollte im Sichtbereich der Benutzer*innen sein, also beispielsweise an einem Ort im Büro platziert sein, an dem viele vorbeigehen. Es sollte keine Interaktion mit dem Dashboard notwendig sein, um die gelieferten Informationen zu verstehen. Damit die Benutzer*innen Gewohnheiten aufbauen können, sollten gleiche Informationen immer an der gleichen Stelle auf dem Dashboard platziert werden. Zu guter Letzt ist nicht zu vernachlässigen, das Dashboard visuell ansprechend zu gestalten, um das Interesse der Benutzerin oder des Benutzers am Betrachten zu steigern. (Janes u. a., 2013, S. 20f.)

2.7 E-Commerce

Da die Feature-Auswertung in erster Linie für Onlineshops durchgeführt wird, wird das Thema *E-Commerce* behandelt und der Aufbau sowie wichtige Komponenten eines Onlineshops beschrieben.

E-Commerce steht für den elektronischen Handel, insbesondere für Verkaufsvorgänge, die über das Internet abgewickelt werden. Als Vertriebsweg kommen Onlineshops zum Einsatz.

Hauptaufgaben eines Onlineshops sind die Produktpräsentation sowie die Abwicklung des Kaufprozesses (Kollewe, 2016, S. 3). Onlineshops unterscheiden sich abhängig von ihrer

Zielgruppe. Privatkunden sind eine deutlich inhomogenere Zielgruppe als gewerbliche Kunden (Kollewe, 2016, S. 15f.) und ihr Kaufverhalten dadurch schwerer vorhersehbar. Um sie besser ansprechen zu können, ist es wertvoll mehr darüber zu erfahren, wie sie den Onlineshop und insbesondere spezielle Features des Shops nutzen. Einer der Marktführer unter den Online-Handel-Lösungen im Jahr 2020 ist laut dem *Gartner Magic Quadrant* die Onlineshop-Software *Magento* (Gartner, Inc., 2020).

Die meisten Onlineshops folgen einem Basis-Layout, an das sich Kunden gewöhnt haben und in dem sie sich intuitiv zurechtfinden (Kollewe, 2016, S. 133). Ein Onlineshop hat zunächst eine Navigation, die dem Kunden ermöglicht, sich zu orientieren und einen Überblick über das Produktsortiment zu verschaffen. Häufig ist in der Navigation auch das Eingabefeld für die Produktsuche integriert, was dem Kunden ermöglicht, spezifische Anfragen zu stellen. (Kollewe, 2016, S. 113) Auch die Startseite soll dem Kunden Orientierung über das Produktsortiment verschaffen (Kollewe, 2016, S. 142) und kann aktuelle, saisonale oder auch speziell auf den Kunden zugeschnittene Angebote zeigen. Zudem gibt es Kategorieseiten, die Produkte bestimmter Warengruppen oder nach einer thematischen Zusammenstellung anzeigen (Kollewe, 2016, S. 150).

Schließlich landet der Kunde über eine der beschriebenen Komponenten oder von einer externen Suchmaschine oder Webseite auf der Produktdetailseite. Diese soll ausführlich über das Produkt informieren, aber auch alle nötigen Informationen zum Kauf, wie Versandkosten, Lieferzeiten oder mögliche Zahlungsarten, bereitstellen (Kollewe, 2016, S. 157). Sie kann auch Produktvorschläge zu ähnlichen Produkten enthalten.

Hat die oder der Kund*in Produkte in den Warenkorb gelegt, ist sie oder er bereit für den Checkout-Prozess. Nun werden die nötigen Informationen für den Kaufabschluss und Versand eingegeben und eine Zahlungsart gewählt. Bestenfalls schließt sie oder er den Kauf ab und die Bestellung wird aufgegeben. Die oder der Shop-Betreiber*in kann sich, nachdem die Zahlungsabwicklung geklärt ist, um den Versand der Waren kümmern. In der Regel sind Onlineshops zur Automatisierung der Versandabwicklung mit einem Warenwirtschaftssystem verbunden.

2.8 Google Analytics

Das Tracking-Tool *Google Analytics* wird vorgestellt, da es wertvolle Informationen für die Bewertung von Features liefern kann. Es sammelt Informationen über Ereignisse auf Webseiten, um die Nutzung dieser besser zu verstehen.

Die Datenerhebung findet hauptsächlich durch die Analyse von Log-Dateien und durch Page Tagging statt (Aden, 2012, S. 39). Die Log-Dateien enthalten in der Regel die IP-Adresse der Nutzenden sowie Angaben darüber, wann und worauf zugegriffen wurde, wie viel von der Seite übertragen wurde und welches Gerät die nutzende Person verwendet (Aden, 2012, S. 40). Durch Page Tagging werden Daten erhoben, indem Skripte auf den zu trackenden Seiten ausgeführt werden, sobald diese aufgerufen werden. Diese Skripte kommunizieren mit Google Analytics und übertragen entsprechende Informationen. (Aden, 2012, S. 43f.) Auf diese Weise können unter anderem Informationen gesammelt werden, wie häufig Komponenten eines Features verwendet und angeklickt werden. Anhand gesetzter Cookies können die Informationen einer oder einem Nutzer*in zugeordnet und dessen Aktivitäten getrackt werden (Aden, 2012, S. 43f.).

Getrackte Daten können in Onlineshops beispielsweise auch E-Commerce-Informationen wie gekaufte Produkte sein (Aden, 2012, S. 45). Dies ist durch das Tracken von E-Commerce-Transaktionen möglich. Ein Kaufabschluss und damit Informationen über gekaufte Produkte können getrackt werden, indem ein Skript auf der Bestellbestätigungsseite eingebunden wird und nach dem Kauf die Information an Google Analytics sendet. (Aden, 2012, S. 152f.) Dies ermöglicht auch die Ermittlung der Conversion-Rate, dem Verhältnis aus der Anzahl abgeschlossener Käufe und aller Zugriffe (Aden, 2012, S. 557). Ebenso können die Gesamtumsätze und der durchschnittliche Bestellwert auf diese Weise kumuliert werden (Aden, 2012, S. 558f.). Letztlich ist auch das Tracking von Kaufabbrüchen nützlich, unter anderem auf welcher Seite im Checkout-Prozess der Kunde ausgestiegen ist, um Informationen zu erhalten, wo es Optimierungspotential gibt (Kollewe, 2016, S. 201).

Google Analytics bietet auch die Möglichkeit das Timing auf Webseiten zu tracken. Zum einen ist es möglich, die Ladezeit spezifischer Ressourcen zu tracken, es kann aber auch gemessen werden, wie lange ein*e Nutzer*in braucht, bevor sie oder er eine bestimmte Aktion ausführt. (Aden, 2012, S. 177) Auf diese Weise kann ein Stück weit festgestellt werden, wie Nutzer*innen auf bestimmte Features und Feature-Änderungen eines Onlineshops reagieren.

3 Anforderungen

Dieses Kapitel definiert die Anforderungen an das zu entwickelnde System.

Anfangs wird das Konzept der Anforderungsbeschreibung erläutert. Anschließend werden das Einsatzgebiet und die Akteure, welche mit dem System interagieren, beschrieben. Es folgen die Anforderungen an die *Einrichtung und Konfiguration*, die *Visualisierung auf Dashboards* und die *Analyse eines Onlineshops*, sowie die *nichtfunktionalen Anforderungen*.

3.1 Struktur der Anforderungen

Jede der folgenden Anforderungen wird zunächst durch eine User Story eingeleitet, um den Mehrwert für die nutzende Person klarzustellen. Anschließend werden die Akzeptanzkriterien aufgelistet, um genauer zu spezifizieren, wann die Anforderung erfüllt ist.

3.1.1 User Story

Eine User Story beschreibt eine konkrete Funktionalität aus Sicht der Anwenderin oder des Anwenders (Wirdemann, 2017, S. 9). Sie bringt in ein bis zwei Sätzen den Kern der Anforderung auf den Punkt (Wirdemann, 2017, S. 10).

Zunächst wird die Rolle der Anwenderin oder des Anwenders genannt, sodass klar ist, wer die Anforderung stellt und für wen sie einen Mehrwert bietet. Anschließend wird beschrieben, welche Funktionalität sich die oder der genannte Anwender*in wünscht.

Eine Erweiterung der User Story, die den Mehrwert für die oder den Anwender*in verdeutlicht, ist das konkrete Benennen des Zwecks. Dazu wird am Ende einer User Story erklärt, warum die oder der Anwender*in sich eine bestimmte Funktionalität wünscht und welche Vorteile sie oder er sich dadurch erhofft.

Abbildung 3.1 zeigt die typische Struktur einer User Story.

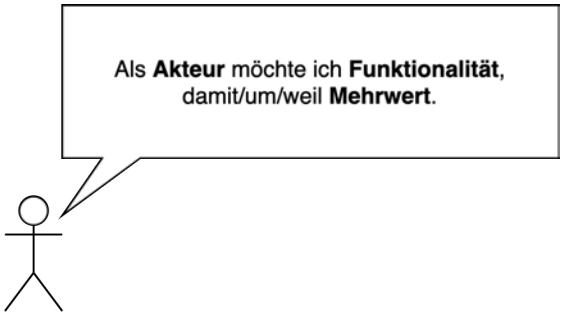


Abbildung 3.1: Typische Struktur einer User Story

3.1.2 Akzeptanzkriterien

Akzeptanzkriterien bestimmen, wann eine Anforderung erfüllt ist und einen Mehrwert liefert (Wirdemann, 2017, S. 52). Für Entwickler*innen sind sie wichtige Anhaltspunkte dafür, was getan werden muss. Sie konkretisieren das Ziel, und geben vor, wann die Bearbeitung der Anforderung abgeschlossen ist (Wirdemann, 2017, S. 53). Anhand der Kriterien kann getestet werden, ob die Anforderung erfüllt ist. Die Akzeptanzkriterien können stichpunktartig formuliert werden.

3.2 Einsatzgebiet und Akteure

Das System kommt dort zum Einsatz, wo Features von Webanwendungen analysiert werden sollen und die dazu benötigten Informationen über Web-APIs bereitgestellt werden. Insbesondere kann das System zur Analyse von Features eines Onlineshops eingesetzt werden.

Die mit dem System agierenden Akteure werden im Folgenden beschrieben.

3.2.1 Entwickler*innen

Entwickler*innen möchten das System mit möglichst geringem Aufwand aufsetzen. Sie wollen sich nicht mit der Funktionsweise des Systems auseinandersetzen müssen. Daher sollten sie vom System durch die nötigen Konfigurationsschritte geführt werden.

Außerdem konfigurieren die Entwickler*innen neu implementierte Features im System, sodass die Features auf dem Dashboard aktuell sind. Auch hier ist besonders wichtig, dass die Konfiguration möglichst unkompliziert ist, um den Entwicklungsprozess nicht auszubremsen und die Akzeptanz für das System zu steigern.

3.2.2 Produktverantwortliche

Produktverantwortliche sind Personen, die Verantwortung für die Leistungssteigerung einer Anwendung, wie einem Onlineshop, tragen und ihren stetigen Fortschritt fördern. Ihnen dienen die Visualisierungen, um aktuelle Stärken und Defizite dieser Anwendung besser zu erkennen und gezielte Schritte zu ihrer Verbesserung einleiten zu können.

3.3 Anforderungen an das System

3.3.1 Einrichtung und Konfiguration

*Entwickler*innen* haben Anforderungen, wenn es darum geht, Dashboards für neue Projekte einzurichten und Features in die Analyse einzubinden.

Einrichtung eines neuen Projekts

Als *Entwickler*in* möchte ich in kurzer Zeit durch Konfiguration ein Dashboard für ein neues Projekt verfügbar machen, damit ich Kennzahlen auf diesem betrachten kann.

Akzeptanzkriterien:

- Konfiguration über eine Weboberfläche möglich.
- Neues Abruf-System lässt sich hinzufügen.
- Neue Entität, die von einem Abruf-System erhalten werden kann, lässt sich hinzufügen.

Konfiguration einer neuen Kennzahl

Als *Entwickler*in* möchte ich mit geringem Aufwand Kennzahlen zu neuen Features auf dem Dashboard anzeigen lassen können, damit ich schon früh nach dem Release des Features dessen Performance verfolgen kann. Diese Tätigkeit wird häufig durchgeführt. Ein geringer Aufwand ist daher besonders wichtig, damit der Entwicklungsprozess nicht verlangsamt wird.

Akzeptanzkriterien:

- Konfiguration über eine Weboberfläche möglich.
- Transformierung von Quell- zu Zieldaten kann eingestellt werden.
- Die Art der Konfiguration ist abstrakt genug, um Kennzahlen aus unterschiedlichsten Quelldaten zu berechnen.

Rohdaten-Übersicht

Als *Entwickler*in* möchte ich eine Übersicht erhalten, welche Daten sich im Data Lake befinden, um zu wissen, welche Analysen ausgeführt werden können.

Akzeptanzkriterien:

- Es ist sichtbar, zu welchen Entitäten Datensätze vorliegen und wie viele.
- Es ist sichtbar, wie aktuell die Datensätze sind.

Voransicht der Rohdaten bei der Konfiguration

Als *Entwickler*in* möchte ich eine Voransicht der Rohdaten erhalten, anhand derer ich die Konfiguration durchführen und an Beispielen validieren kann, um bestimmen zu können, welche Informationen zu einem Feature ausgewertet werden sollen.

Akzeptanzkriterien:

- Die Ansicht von Quelldatensätzen ermöglicht eine bessere Vorstellung über verfügbare Informationen.

- Werte aus den abgerufenen Daten werden verwendet, um das Ergebnis der Transformation zu veranschaulichen.

Automatische Aktualisierung

Als *Entwickler*in* möchte ich, dass die Daten in regelmäßigen Abständen automatisch abgerufen und verarbeitet werden, damit die angezeigten Informationen stets aktuell sind und ich keinen zusätzlichen Aufwand habe.

Es ist abzuwägen, wie häufig die Aktualisierung mindestens stattfinden sollte, um einen gewinnbringenden Effekt zu erzielen. Da sich dies für verschiedene Anwendungen unterscheiden kann, ist es sinnvoll, wenn das Aktualisierungsintervall konfigurierbar ist.

Akzeptanzkriterien:

- Das Aktualisierungsintervall kann für jede Zieldaten-Entität per Konfiguration festgelegt werden.
- Die für diese Zieldaten-Entität notwendigen Quelldaten werden in diesem Intervall abgerufen.
- Die Verarbeitung und Aktualisierung der Zieldaten erfolgen automatisch.

3.3.2 Visualisierung auf Dashboards

*Entwickler*innen* und *Produktverantwortliche* haben bestimmte Anforderungen an das Dashboard. Sie möchten schnell einen Überblick über die Features und bei Bedarf nähere Informationen erhalten.

Aktualität der Daten

Als *produktverantwortliche Person* möchte ich, dass die Daten auf den Dashboards aktuell sind, sodass ich das momentane Geschehen kenne und sinnvolle Entscheidungen treffen kann.

Akzeptanzkriterien:

- Die Daten werden kontinuierlich aktualisiert.

- Das Dashboard wird in regelmäßigen Abständen neu geladen.

Umfang der Daten

Als *produktverantwortliche Person* möchte ich auf dem Dashboard nur Informationen gezeigt bekommen, die mir einen Nutzen bieten und für meine Entscheidungen notwendig sind, um Übersichtlichkeit zu bewahren.

Akzeptanzkriterien:

- Es werden nur die Informationen angezeigt, welche konfiguriert wurden.

3.3.3 Nichtfunktionale Anforderungen

Neben den funktionalen Anforderungen gibt es Eigenschaften, die erfüllt werden sollen, um die Wartbarkeit, Effizienz, Zuverlässigkeit, Benutzbarkeit und Sicherheit des Systems zu gewährleisten.

Weitere Systeme einheitlich einbinden

Weitere Abruf-Systeme sollen auf eine einheitliche Art eingebunden werden können, um Daten aus neuen Abruf-Systemen abfragen zu können. Der Code für neue Abruf-Systeme soll als Plugin erstellt werden können. Dieses soll dynamisch eingebunden werden, ohne dass Anpassungen der Kernsoftware notwendig sind.

Kein mehrfacher Abruf von Daten

Nur neue oder veränderte Daten sollen abgerufen werden, um den Traffic und Speicherbedarf gering zu halten, sowie die Abfragen schnell durchführen zu können. Es wird festgestellt, welche Daten sich verändert haben und nur diese abgerufen. Bei Unterbrechungen des Abrufs dürfen keine Inkonsistenzen entstehen.

Andere Systeme nicht zu stark belasten

Systeme, von denen Daten abgefragt werden, dürfen nicht so sehr ausgelastet werden, dass sie in ihrer Funktionalität eingeschränkt werden. Das Abrufintervall spielt dabei eine Rolle. Wird das Intervall sehr groß, besteht die Gefahr, dass große Mengen von Daten auf einmal abgerufen werden, was zu einer erhöhten Last auf die Anwendung führen kann. Um dies zu verhindern, soll beim Abrufen sichergestellt werden, dass die abzurufende Datenmenge nicht zu groß und gegebenenfalls in kleinere Abfragen aufgeteilt wird.

Verständliche, effektive und zielgruppengerechte Darstellung der Daten

Die Daten sollen so dargestellt werden, dass sie gut verständlich sind und *Produktverantwortliche* oder *Entwickler*innen* effektiv mit ihnen arbeiten können.

Datenschutz und Datensicherheit

Wenn möglich, sollten die gesammelten Daten von den Personen getrennt werden, sodass es nicht mehr möglich ist, zu erkennen, wer hinter dem Datensatz steht. Die anonymen Daten können dann für den Geschäftszweck verwendet werden. (Launix, 2016)

Zur Gewährleistung der Datensicherheit dürfen Daten nur über verschlüsselte Verbindungen übertragen werden. Die Systeme, auf denen die Daten gespeichert sind, müssen mit aktueller Software laufen und dürfen nur für autorisierte Personen zugänglich sein.

3.4 Analyse eines Onlineshops

Zum Testen des Systems werden Anwendungsfälle für die Analyse eines Onlineshops konfiguriert.

Auswertung von Zahlungsarten

Zur Auswertung der Zahlungsarten wird die Nutzungshäufigkeit und die Höhe der Umsätze pro Zahlungsart analysiert.

Umsatz in Zusammenhang mit Releases

Die Umsatzentwicklung und Bestellhäufigkeit wird im zeitlichen Zusammenhang mit den Releases dargestellt. Die Beschreibung der Releases wird angezeigt, damit festgestellt werden kann, welches Feature veröffentlicht wurde.

Umsatz in Zusammenhang mit einem Feature

Die Umsatzentwicklung und Bestellhäufigkeit wird nach Bestellungen, die in Zusammenhang mit einem bestimmten Feature stehen, gefiltert.

4 Architektur

In diesem Kapitel wird die Architektur des Systems beschrieben.

Zunächst wird erläutert, wie die Anforderungen umgesetzt werden sollen. Es folgen die unterschiedlichen Sichten auf die Architektur des Systems. Anschließend werden die Komponenten des Systems beschrieben. Im nächsten Abschnitt wird die Struktur der Konfiguration definiert und die Funktionen der Konfigurationsoberfläche erläutert. Abschließend wird dargelegt, wie das System hochgefahren werden soll und wie es dokumentiert wird.

Der Service, welcher die Konfigurationsoberfläche zur Verfügung stellt und die Daten transformiert, wird künftig mit dem Namen *FeatDisco* betitelt. *FeatDisco* setzt sich aus den Begriffen Feature und Discovery zusammen und steht für den Erkenntnisgewinn durch die Auswertung der Feature-Metriken.

4.1 Umsetzung der Anforderungen

In diesem Abschnitt wird dargelegt, wie die Anforderungen aus Kapitel 3 umgesetzt werden.

Einrichtung eines neuen Projekts

Damit die Einrichtung neuer Projekte zügig erfolgen kann, muss die Konfiguration schnell zugänglich sein. Dies wird dadurch erfüllt, dass die Konfiguration über eine Weboberfläche durchgeführt wird, die in jedem Browser aufgerufen werden kann. Hierüber lassen sich innerhalb kurzer Zeit neue Abruf-Systeme und Abruf-Entitäten hinzufügen (vgl. *4.3.2 Datenabruf und Speicherung der Rohdaten*).

Über die HTTP-API von Grafana ist es möglich, eine neue Datenquelle und ein neues Dashboard in Grafana automatisch anzulegen (vgl. *2.5.5 HTTP-API*).

Konfiguration einer neuen Kennzahl

Die Konfiguration der Auswertung von Daten kann ebenfalls über eine Weboberfläche durchgeführt werden, was den Aufwand bei häufiger Durchführung gering hält (vgl. *4.3.3 Konfiguration der Transformation*). Sie ist abstrakt gehalten, sodass unterschiedliche Quelldaten unterstützt werden.

Um zielgerichtet festzustellen, welche Daten benötigt werden, um ein neues Feature zu messen, kann das GQM-Modell angewandt werden. Vom anvisierten Ziel, über die Fragestellungen gelangt man zu den Metriken, die für die neue Auswertung verwendet werden können (vgl. *2.6.1 Auswahl der Daten*).

Nachdem die Darstellungsweise festgelegt wurde (vgl. *2.6.2 Wahl der Visualisierungstechnik*), kann diese auf dem Grafana-Dashboard eingestellt werden.

Rohdaten-Übersicht

Um eine Übersicht über die Daten im Data Lake zu erhalten, können *Entwickler*innen* die Übersichtsseite der Konfigurationsoberfläche betrachten (vgl. *4.5 Funktionen der Konfigurationsoberfläche*). Diese bietet ihm die Möglichkeit zu erfahren, zu welchen Entitäten es wie viele Datensätze gibt und wie aktuell diese sind.

Voransicht der Rohdaten bei der Konfiguration

Um die Konfiguration der Transformation anschaulicher zu machen und um konfigurierte Werte validieren zu können, kann über einen Button auf der eingerichteten Ziel-Entität eine Vorschau eines Datensatzes vor und nach der Transformation angezeigt werden. Dazu wird ein Datensatz anhand der konfigurierten Werte transformiert, aber noch nicht abgespeichert. Dies ermöglicht den *Entwickler*innen*, verschiedene Konfigurationen auszuprobieren.

Automatische Aktualisierung

Für die automatische Aktualisierung werden die Aktionen, die auch manuell gestartet werden können, in bestimmten Zeitintervallen ausgeführt. Die Zeit, wann und wie häufig die Aktualisierung stattfindet, kann für jede Ziel-Entität über die Konfigurationsseite

festgelegt werden. Es wird dann ein Cronjob eingerichtet, welcher in diesem Zeitintervall nacheinander die benötigten Rohdaten abruft und anschließend die Transformation der Daten startet. Nachdem eine Ziel-Entität einmal eingerichtet wurde, müssen *Entwickler*innen* somit keine weiteren Aktionen ausführen, um die Visualisierungen auf dem Dashboard aktuell zu halten.

Aktualität der Daten

Durch die automatische Aktualisierung der Daten werden auch auf dem Dashboard laufend aktuelle Daten angezeigt. In Grafana wird eingestellt, in welchem Zeitintervall die Visualisierungen aktualisiert werden.

Umfang der Daten

Durch die Transformation der Daten in festgelegte Ziel-Entitäten werden die Daten auf die relevanten Informationen begrenzt. Indem die Auswahl der Daten auf dem erwarteten Nutzen basiert und der Fokus bei der Visualisierung auf das Geschäftsziel gelegt wird (vgl. *2.6 Informationsdesign*), ist die Übersichtlichkeit des Dashboards gewährleistet und die für Entscheidungen relevanten Informationen können abgelesen werden.

Weitere Systeme einheitlich einbinden

Weitere Systeme lassen sich als Plug-Ins einbinden (vgl. *4.3.1 Abruf-System-Typen*). Dazu muss der Code der Kernsoftware nicht verändert werden. Der neue Code für das Plug-In wird in einem dafür vorgesehenen Verzeichnis abgelegt und das neue System kann direkt über die Konfigurationsoberfläche eingesetzt werden (vgl. *4.3.2 Datenabruf und Speicherung der Rohdaten*).

Kein mehrfacher Abruf von Daten

Es werden nur Daten abgerufen, die neu sind oder sich verändert haben (vgl. *2.3 Change Data Capture*). Wie dies umgesetzt wird, ist abhängig von der API, von der die Daten abgefragt werden. Am Beispiel der Bestelldaten von Magento, kann ein Zeitstempel verwendet werden, der angibt, wann die Daten zuletzt aktualisiert wurden. Über die API

kann danach gefiltert werden, dass nur Bestelldaten zurückgegeben werden, die nach einem bestimmten Zeitpunkt geändert wurden. Nach dem Abruf wird der Zeitpunkt gespeichert, bis zu welchem die Daten abgerufen wurden. So werden Inkonsistenzen vermieden, wenn der Abruf unterbrochen wird. Beim nächsten Starten des Abrufs wird an der Stelle fortgesetzt, an welcher der Abruf abgebrochen oder abgeschlossen wurde.

Andere Systeme nicht zu stark belasten

Die Belastung der Abruf-Systeme steigt, wenn große Datenmengen in kurzer Zeit angefragt werden. Um eine zu große Belastung zu vermeiden, selbst wenn der Abruf seltener durchgeführt wird, muss ein maximales Abruf-Intervall festgelegt werden (vgl. *4.3.2 Datenabruf und Speicherung der Rohdaten*). Abhängig von der anfallenden Menge an Daten pro Zeit sollte dieses Intervall kleiner gewählt werden. Es kann für die Abruf-Entitäten festgelegt werden. Ist das abzufragende Zeitintervall größer als das festgelegte maximale Intervall, wird der Abruf auf mehrere Abfragen aufgeteilt, zwischen denen eine kurze Zeit abgewartet wird.

Verständliche, effektive und zielgruppengerechte Darstellung der Daten

Um die Daten so darzustellen, dass sie effektiv genutzt werden können, wird die Visualisierungstechnik abhängig von den spezifischen Anforderungen der Zielgruppe gewählt (vgl. *2.6.2 Wahl der Visualisierungstechnik*).

Datenschutz und Datensicherheit

Bei der Transformation der Daten werden nur die für den Geschäftszweck benötigten Daten übernommen. In der Regel werden keine Informationen in den Daten benötigt, welche den Bezug zu einer Person erkennbar machen. Falls solche Bezüge erforderlich sind, werden die sensiblen Daten durch Ersetzungen anonymisiert.

Daten werden nur über verschlüsselte Verbindungen übertragen (vgl. *4.3.5 Zugriff über sichere Verbindung*). Der Zugriff auf das System ist geschützt und die Zugangsdaten sind nur autorisierten Personen bekannt. Die Software auf dem System ist aktuell.

4.2 Architektursichten

Die Architektursichten orientieren sich am arc42 Template (Starke und Hruschka, 2005). Zunächst vermittelt die *Kontextabgrenzung* einen Überblick über das System und die Akteure. Die *Laufzeitsicht* gibt Einblick in den Ablauf der Prozesse im System und beteiligter Komponenten. Die *Verteilungssicht* zeigt die technische Infrastruktur auf.

4.2.1 Kontextabgrenzung

Abbildung 4.1 zeigt die Kontextabgrenzung des Systems, welche dieses von seinen Nachbarsystemen und den Benutzerrollen abgrenzt und die externen Schnittstellen festlegt (Starke und Hruschka, 2005). Zentrale Rolle spielt der Onlineshop, mit dem die potenziellen Käufer*innen interagieren. *Entwickler*innen* implementieren neue Features für den Onlineshop. In manchen Fällen müssen zusätzliche Metriken zu diesen Features explizit über die API verfügbar gemacht werden. Sobald ein Feature im Onlineshop Anwendung findet, wird die Auswertung für dieses über die Konfigurationsoberfläche von *FeatDisco* von einer oder einem *Entwickler*in* konfiguriert. *FeatDisco* teilt daraufhin dem Dashboard-System mit, dass die entsprechenden von *FeatDisco* bereitgestellten Daten vom Dashboard-System auf die gewünschte Weise visualisiert werden sollen. Dieses ruft anschließend die Daten von *FeatDisco* ab, um die Visualisierungen aktuell zu halten. Schließlich liest die oder der Shop-Betreiber*in beziehungsweise *Produktverantwortliche* die Feature-Adaption auf dem Dashboard ab.

4.2.2 Laufzeitsicht

Laufzeitsichten verdeutlichen die zeitlichen Abläufe zwischen den beteiligten Akteuren und Systemkomponenten anhand von konkreten Szenarien der wichtigsten Abläufe (Starke und Hruschka, 2005).

Abbildung 4.2 zeigt den Ablauf der Implementierung und Konfiguration eines neuen Features bis hin zur Anzeige der Visualisierung auf dem Dashboard. Nachdem ein*e *Entwickler*in* die Implementierung eines neuen Features in einem Onlineshop abgeschlossen hat und sichergestellt hat, dass die zugehörigen Informationen über eine Web-API abrufbar sind, konfiguriert die Person den Abruf und die Auswertung der Informationen über die Konfigurationsoberfläche von *FeatDisco*. Daraufhin legt *FeatDisco* über die API

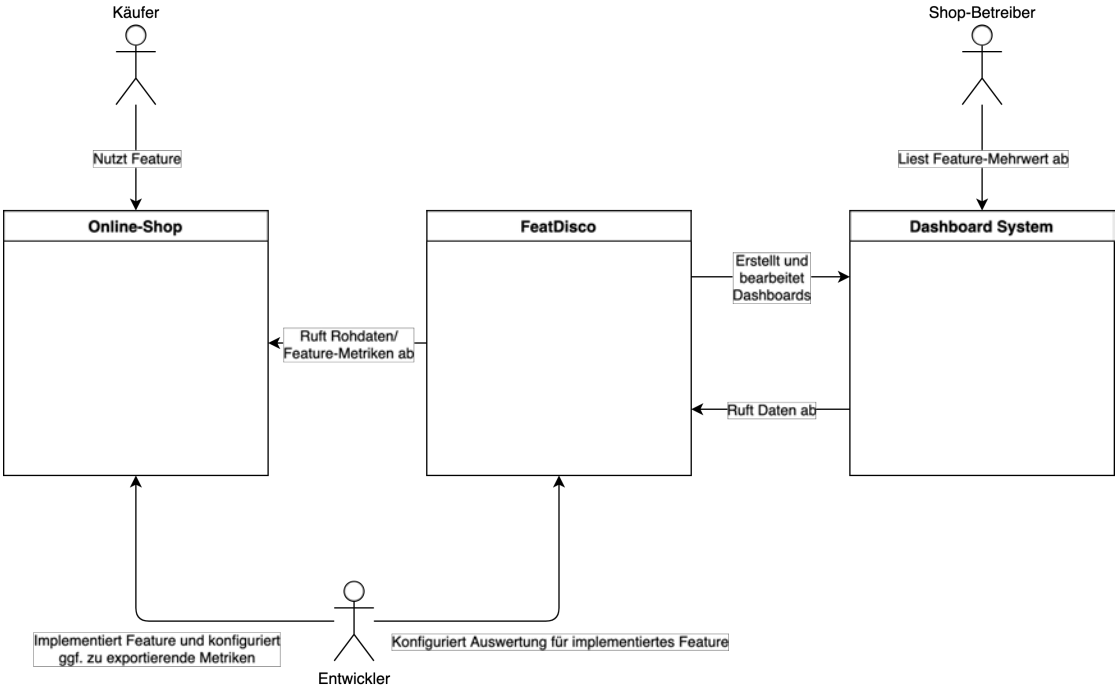


Abbildung 4.1: Architektursicht Kontextabgrenzung

des Dashboard-Systems die Datenquelle und Visualisierung für die soeben konfigurierten Daten an. *FeatDisco* beginnt nun, kontinuierlich die Daten vom Onlineshop und weiteren Abruf-Systemen abzurufen. Letztere sind aus Gründen der Anschaulichkeit in der Abbildung nicht aufgeführt. Daten, die anhand der Konfiguration vorbereitet wurden, werden abgespeichert. Das Dashboard-System ruft die vorbereiteten Daten kontinuierlich ab und hält die Visualisierungen aktuell. Nun kann ein*e Shop-Betreiber*in, Entwickler*in oder Projektverantwortliche*r das Dashboard einsehen. Der Übersichtlichkeit halber wurden nicht alle Rollen bei diesem Schritt in der Abbildung aufgeführt.

4.2.3 Verteilungssicht

Abbildung 4.3 zeigt die Verteilungssicht des Systems. Diese Sicht zeigt die technische Infrastruktur, auf der die Komponenten des Systems ausgeführt werden.

Die Anwendung zum Abruf und zur Vorbereitung der Daten läuft in einem Node.js Docker Container. Ein weiterer Docker Container, in dem Elasticsearch läuft, dient der Bereithaltung der vorbereiteten Daten. Das Dashboard-System Grafana läuft in einem dritten Docker Container. Die Container werden gemeinsam mit Docker Compose gestartet.

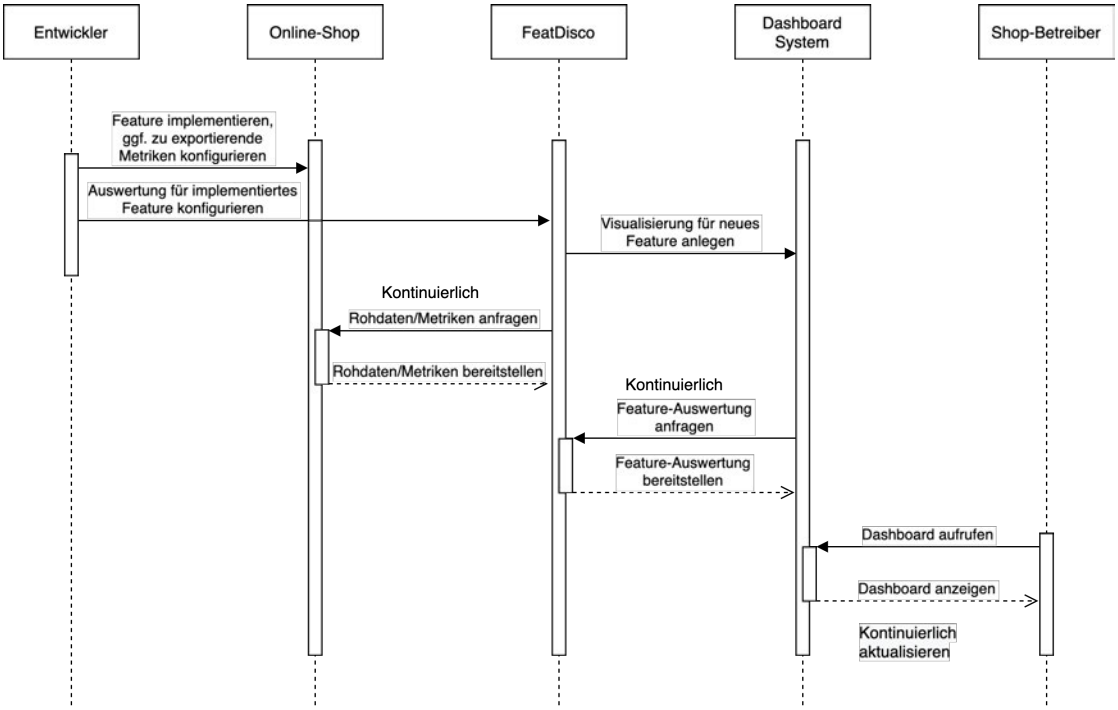


Abbildung 4.2: Architektursicht Laufzeitsicht

Ein Web-Browser kann auf den Onlineshop, die Weboberfläche von *FeatDisco* und das Dashboard-System zugreifen. Die Weboberfläche von *FeatDisco* gibt Änderungen an die Konfigurationskomponente weiter. Dadurch verändert sich das Verhalten beim Abruf sowie der Transformation und dem Mapping. Darüber hinaus kann durch Änderung der Konfiguration die Dashboard-API und Datenquellen-API des Dashboard-Systems angesprochen werden, um Datenquellen und Dashboards anzulegen oder zu verändern.

Der Abruf findet über einen Adapter für das jeweilige Abruf-System statt. Die abgerufenen Daten werden als Rohdaten in einem *Data Lake* abgelegt. Die Transformation und das Mapping nutzen diese Rohdaten und verarbeiten sie gemäß der Konfiguration. Die Daten werden anschließend in *Elasticsearch* gespeichert.

Das Dashboard-System kann, nach Anlegen der Datenquelle über die API, die Daten aus Elasticsearch abrufen und entsprechende Visualisierungen erzeugen.

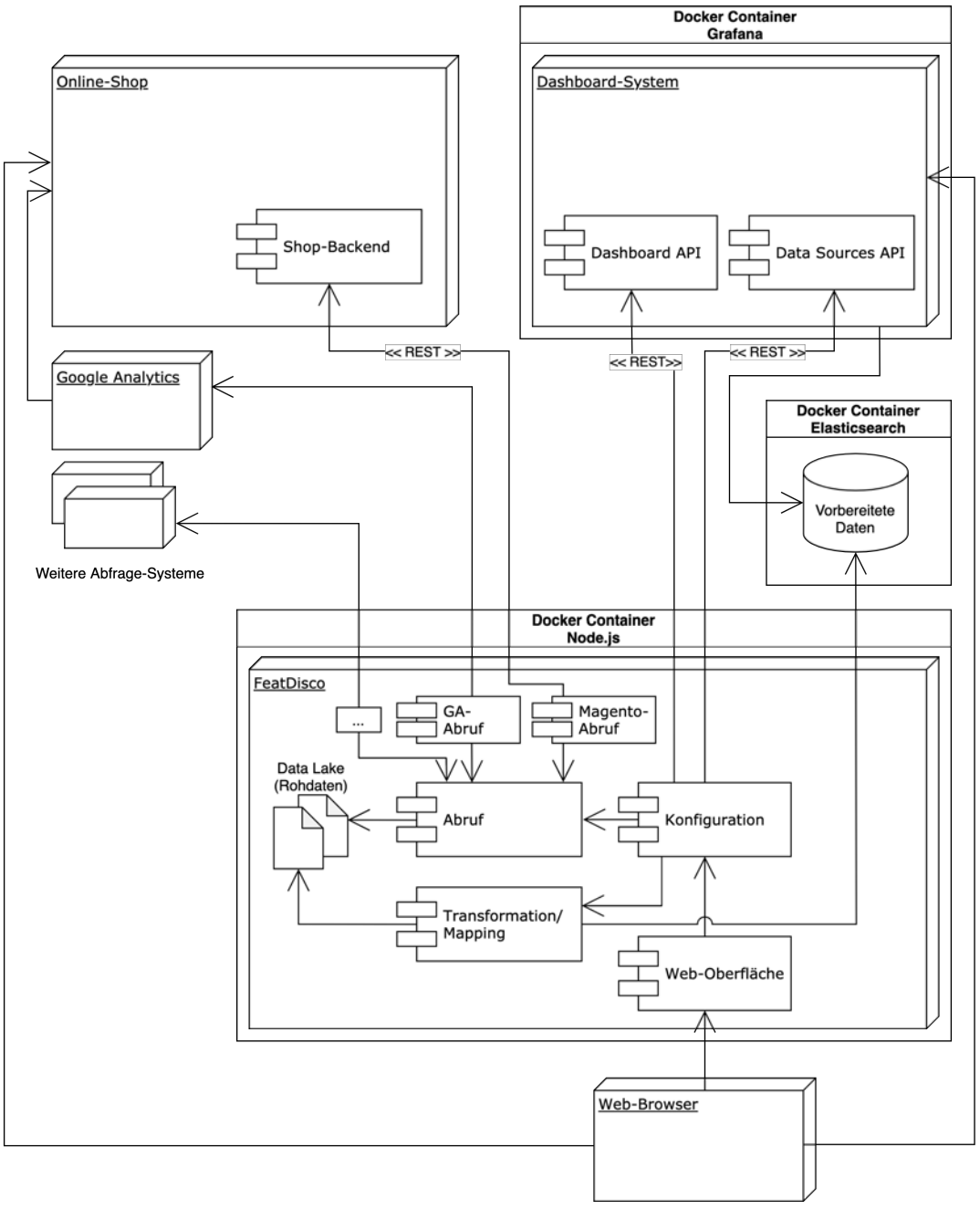


Abbildung 4.3: Architektursicht Verteilungssicht

4.2.4 Data Pipeline

Abbildung 4.4 veranschaulicht den Datenfluss als *Data Pipeline* von den Rohdaten bis zur ausgewerteten und visualisierten Form. Zunächst werden die Daten von den zu untersuchenden Systemen abgerufen. Dazu werden Anfragen an die jeweiligen Web-APIs gestellt. Die Daten werden nach dem *Data Lake*-Prinzip in ihrer ursprünglichen Form gespeichert. Über die Rohdaten im *Data Lake* wird ein Index aufgebaut, sodass Daten daraus abgefragt und Informationen über den Datenbestand bereitgestellt werden können. Nun werden die für die Visualisierung benötigten Daten entsprechend einer vorgegebenen Konfiguration transformiert, auf bestimmte Datentypen abgebildet und in einen *Elasticsearch* Cluster transferiert. Schließlich werden die Daten von einer Dashboard-Instanz aus Elasticsearch abgerufen und zur Analyse dargestellt.

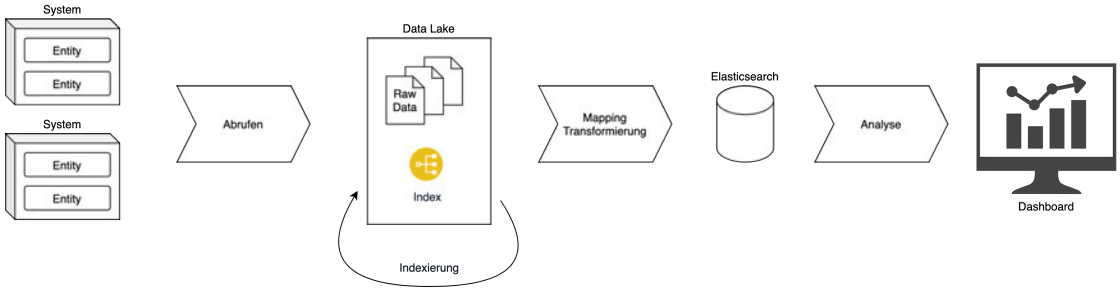


Abbildung 4.4: Data Pipeline

4.3 Komponenten

Das System lässt sich in übergeordnete Komponenten einteilen, die unterschiedliche Aufgaben haben.

4.3.1 Abruf-System-Typen

Systeme, von denen Daten abgefragt werden sollen, können beispielsweise Onlineshop-Systeme wie Magento, Trackingtools wie Google Analytics oder Projektmanagement-Anwendungen wie Jira sein. Jeder dieser System-Typen hat seine Eigenarten, was die Abfrage der Daten angeht. Die spezifischen Implementierungen für die unterschiedlichen Abruf-System-Typen werden daher als Adapter eingesetzt, damit problemlos weitere System-Typen hinzugefügt werden können. Das Laden dieser Implementierungen

geschieht dynamisch, sodass sie nicht zusätzlich registriert werden müssen. Sobald der Code für einen Abruf-System-Typ im dafür vorgesehenen Verzeichnis erstellt wurde, können Abruf-Systeme von diesem Typ konfiguriert werden.

4.3.2 Datenabruf und Speicherung der Rohdaten

Damit Daten abgerufen werden, muss es Implementierungen für die abzufragenden System-Typen geben (vgl. *4.3.1 Abruf-System-Typen*). Per Konfiguration auf einer Weboberfläche kann ein neues Abruf-System von einem der verfügbaren System-Typen hinzugefügt werden. Die Struktur der Konfiguration für die Abruf-Systeme wird in Unterabschnitt 4.4.1 beschrieben.

Gespeichert wird die Konfiguration im JSON-Format. Beim Aufruf der Konfigurationsseite werden aus den JSON-Daten Formularfelder erzeugt, die der Hierarchie in den JSON-Daten entsprechen. Für Abruf-Systeme werden Karten angelegt, auf denen die Formularfelder platziert sind. Auf diesen Karten werden auch weitere Karten für die Abruf-Entitäten platziert, die die Formularfelder für die spezifische Konfiguration der jeweiligen Abruf-Entität enthält.

Die Namen der Formularfelder werden so gewählt, dass sie beim Speichern der Konfiguration wieder in der gleichen Struktur in JSON-Format gesammelt werden können. Das Feld für den Namen des ersten Abruf-Systems erhält beispielsweise den Namen „extract.systems.system1.name“ und das Feld für den Namen der ersten Abruf-Entität des ersten Abruf-Systems den Namen „extract.systems.system1.entities.entity1.name“. Für eine beispielhafte Konfiguration eines Abruf-Systems, wie in Quellcode 4.1, soll die Konfigurationsoberfläche aussehen wie in Abbildung 4.5.

```
1  {
2      "projektname_magento": {
3          "type": "magento",
4          "host": "projecturl.com",
5          "port": 443,
6          "username": "someone",
7          "password": "secret",
8          "totp_secret": "thisisthetotpsecret",
9          "entities": [
10             {
```

```
11                  "name": "orders",
12                  "index_key": "increment_id",
13                  "date_key": "updated_at",
14                  "filter_date_key": "updated_at",
15                  "data_path": "items",
16                  "fields": "",
17                  "start_from": "2021-01-01␣00:00:00",
18                  "only_to": "",
19                  "max_extract_interval": "1d"
20              },
21              {
22                  "name": "paymentrequests",
23                  "index_key": "entity_id",
24                  "date_key": "created_at",
25                  "filter_date_key": "main_table.created_at",
26                  "data_path": "entity_id,created_at,order_id",
27                  "fields": "",
28                  "start_from": "2021-01-01␣00:00:00",
29                  "only_to": "",
30                  "max_extract_interval": "1d"
31              }
32          ]
33      }
34  }
```

Quellcode 4.1: Beispiel für die JSON-Konfiguration eines Abruf-Systems

Die Konfiguration kann auf der Konfigurationsoberfläche editiert werden. Es können Abruf-Systeme und Abruf-Entitäten über das Kreuz entfernt werden und neue über die Buttons hinzugefügt werden. Beim Hinzufügen eines neuen Abruf-Systems wird über ein Auswahlfeld der System-Typ ausgewählt. Die über dieses Auswahlfeld verfügbaren System-Typen werden aus den Konfigurationsdateien der implementierten Abruf-System-Typen zusammengeführt.

Wurde die Konfiguration aus dem Formular der Konfigurationsoberfläche eingelesen, kann der Abruf ausgeführt werden. Dazu wird, anhand des Abruf-System-Typs, die entsprechende Implementierung verwendet. In der Regel wird anfangs ein Token für den Zugriff abgerufen, sofern der vorherige nicht mehr gültig ist, um anschließend die eigentliche Ressource anzufragen. Wenn die Nutzdaten erfolgreich zurückgegeben wurden, werden diese in ihrem Rohzustand abgespeichert und der Speicherort mit der eindeutigen ID im

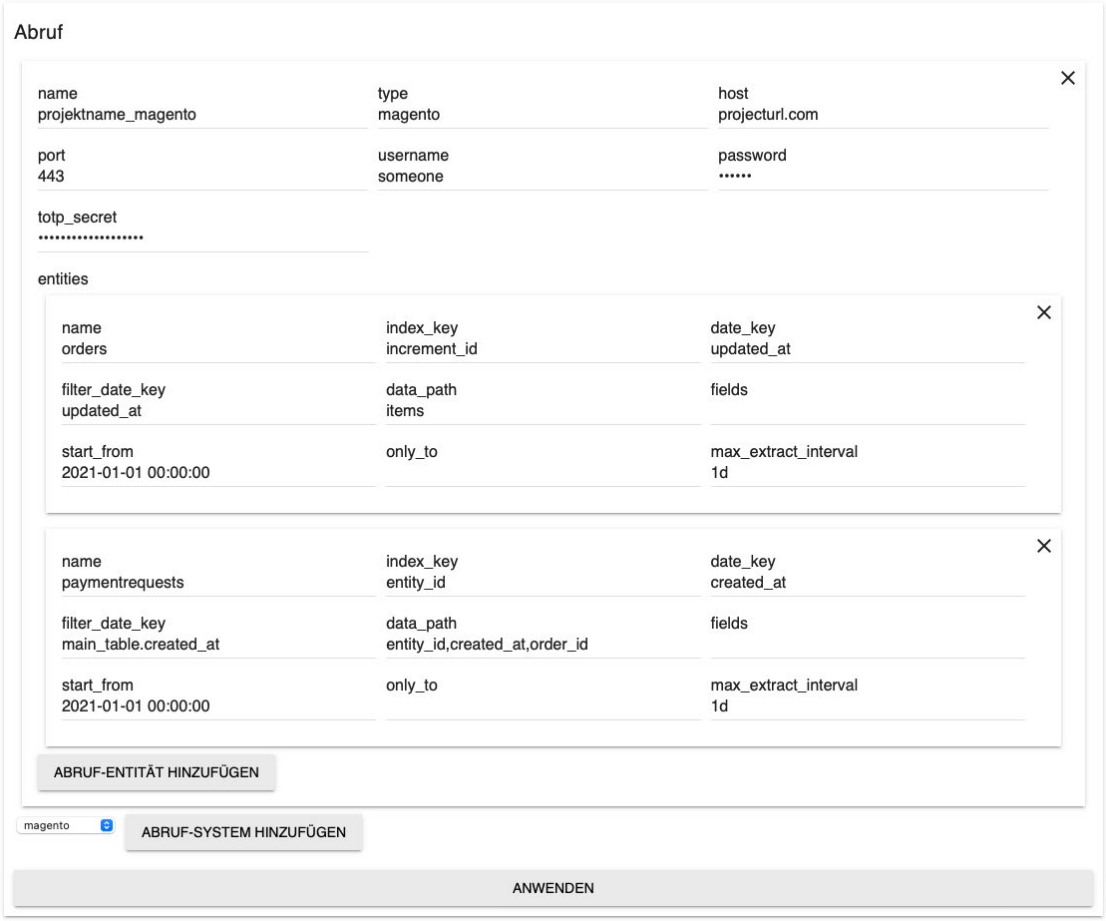


Abbildung 4.5: Beispiel für die Konfiguration eines Abruf-Systems auf der Konfigurationsoberfläche

Index registriert. Im Index wird auch festgehalten, bis zu welchem Zeitpunkt die Daten abgerufen wurden, sodass der nächste Abruf an dieser Stelle fortgesetzt werden kann. Die Daten stehen nun zur Transformation bereit.

4.3.3 Konfiguration der Transformation

Abgerufene Daten stehen zur Transformation zur Verfügung. Es kann konfiguriert werden, welche Felder der Daten übernommen und auf welche Feldnamen und Typen sie gemappt werden.

Daten aus verschiedenen Quellen können zusammengeführt werden. Dazu werden die Felder festgelegt, die den eindeutigen Schlüssel enthalten, über den die Daten verknüpft werden können. Es wird ein Standardwert festgelegt, der übernommen wird, wenn es kein Schlüssel-Pendant aus der anderen Quelle gibt.

Für die transformierten Daten werden Dokumente in Elasticsearch abgelegt oder aktualisiert. Gibt es noch keinen Elasticsearch-Index für die Entität, wird dieser zunächst erstellt. Dabei wird ein Mapping festgelegt, welche Datentypen die Felder in Elasticsearch erhalten. Die Datentypen werden aus dem zuvor konfigurierten Mapping übernommen.

Abbildung 4.6 zeigt die Konfiguration einer Ziel-Entität auf der Konfigurationsoberfläche. Es können neue Ziel-Entitäten hinzugefügt werden. Für diese können zunächst Quell-Entitäten festgelegt werden. Im nächsten Schritt werden die Felder der Ziel-Entität festgelegt. Sie haben einen Namen und enthalten den Datentyp, den das Feld erhalten soll. Bei einigen Datentypen muss das Format, in dem die Quell-Daten vorliegen, angegeben werden, um sie transformieren zu können. Auch die Funktion, nach der die Daten transformiert werden sollen, wird angegeben. Abschließend werden die Feld-Quellen festgelegt, aus denen die zu transformierenden Daten stammen.

4.3.4 Einrichten der Dashboards

Nachdem die Transformation konfiguriert wurde, wird über die Datenquellen-API von Grafana eine neue Datenquelle angelegt. Somit sind die Daten, welche in Elasticsearch abgelegt werden, direkt in Grafana verwendbar.

4.3.5 Zugriff über sichere Verbindung

Damit der Zugriff, sowohl auf das Dashboard-System als auch auf die Konfigurationsoberfläche, über eine sichere Verbindung erfolgt, wird ein Proxyserver eingesetzt. Hierfür kommt die Webserver-Software nginx zum Einsatz. nginx wird so konfiguriert, dass alle unsicheren Verbindungen auf verschlüsselte Verbindungen über das HTTPS-Protokoll umgeleitet werden. Ist eine sichere Verbindung aufgebaut, leitet nginx die Ressourcen vom Dashboard-System und der Konfigurationsoberfläche weiter.

Eine Firewall beschränkt den Zugriff, sodass nur die Ports von außen aufgerufen werden können, welche vom Proxyserver bedient werden. So wird sichergestellt, dass kein unbefugter Zugriff über unsichere Verbindungen auf die Systeme möglich ist.

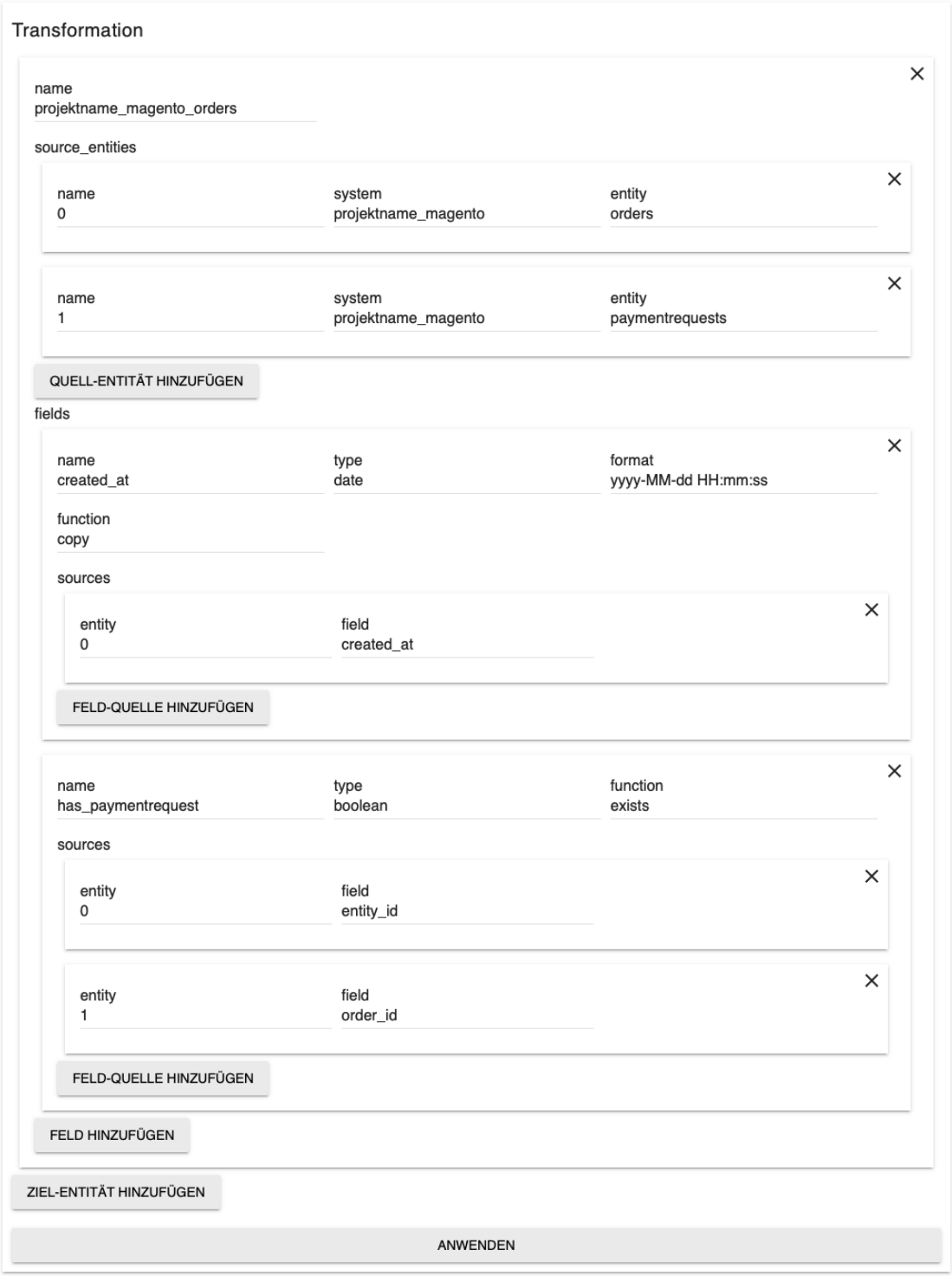


Abbildung 4.6: Beispiel für die Konfiguration einer Ziel-Entität auf der Konfigurationsoberfläche

4.4 Struktur der Konfiguration

In diesem Abschnitt wird die Struktur der Konfiguration beschrieben, wie sie auf der Konfigurationsoberfläche von *FeatDisco* Anwendung findet.

4.4.1 Abruf der Daten

Es gibt drei grundlegende Konfigurations-Entitäten. Diese sind das Projekt, das Abruf-System und die Abruf-Entität. Eine Übersicht über die Konfigurations-Entitäten bietet Abbildung 4.7. Im Folgenden werden die Konfigurations-Entitäten näher beschrieben.

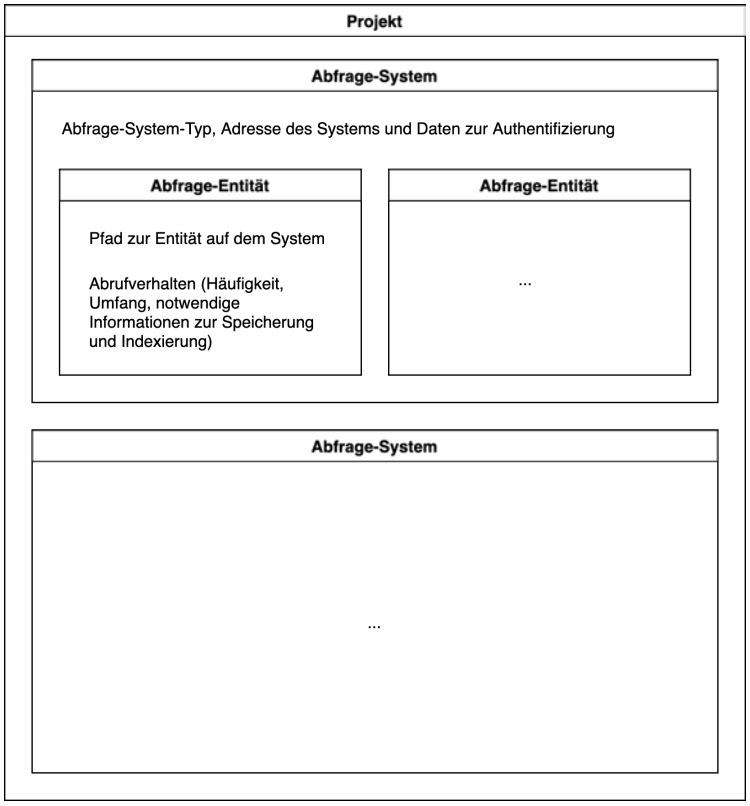


Abbildung 4.7: Konfigurations-Entitäten

Übergeordnet ist das Projekt. Ein Projekt umfasst die Gesamtheit der Systeme, die zur Untersuchung einer Webanwendung wie einem Onlineshop gehören.

Für ein Projekt werden ein oder mehrere Abruf-Systeme konfiguriert. Diese müssen von einem implementierten *Abruf-System-Typ* sein. Darüber hinaus werden für ein Abruf-

System Informationen für den Verbindungsaufbau festgelegt. Dies sind in der Regel der Host und Port des Servers sowie Benutzername und Passwort und bei manchen Systemen weitere Informationen zur Authentifizierung.

Für ein Abruf-System werden eine oder mehrere Abruf-Entitäten konfiguriert. Eine Abruf-Entität spiegelt eine abrufbare Information von einem Abruf-System wider. Diese besitzt einen Pfad auf dem Abruf-System. Für jede Abruf-Entität wird das Abrufverhalten festgelegt. Es wird konfiguriert, wie häufig und für welchen Zeitraum Daten für diese Entität abgerufen werden und wie die Daten abgespeichert werden. Dazu werden nur die notwendigen Informationen zur Struktur der Daten konfiguriert, welche benötigt werden, um die Daten später weiterzuverarbeiten. Dies sind häufig die Namen der Felder, welche die eindeutige ID oder den zeitlichen Bezug halten, anhand derer die Daten im nächsten Schritt aus dem Data Lake abgerufen werden können. Durch den Zeitbezug können die Daten außerdem beim Abruf gefiltert werden, um nur neue und veränderte Daten abzurufen (vgl. *2.3 Change Data Capture*). Zudem können die Daten anhand dieses Feldes auf einer Zeitachse dargestellt werden.

4.4.2 Transformation und Mapping

Für die Weiterverarbeitung der Rohdaten werden Ziel-Entitäten konfiguriert. Für eine Ziel-Entität wird festgelegt, aus welchen Feldern welcher Abruf-Entitäten Daten übernommen werden. Es können auch Daten aus unterschiedlichen Abruf-Entitäten zu einem Feld der Ziel-Entität durch eine Funktion zusammengeführt werden. Zum Beispiel kann für die Ziel-Entität ein Feld angelegt werden, das einen Wahrheitswert enthält, ob es in der zweiten Abruf-Entität einen Eintrag mit der gleichen ID gibt wie in der ersten.

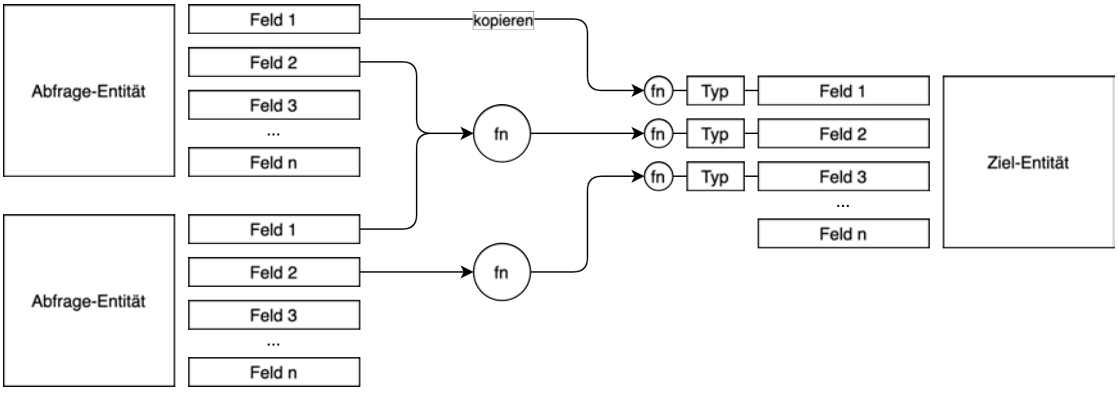


Abbildung 4.8: Daten Transformation

Auch wenn Daten von einem Feld einer Abruf-Entität auf ein Feld der Ziel-Entität übertragen werden, kann der Wert durch eine Funktion transformiert werden. Die Möglichkeiten der Daten-Transformation werden in Abbildung 4.8 veranschaulicht.

Für jedes Feld der Ziel-Entität wird ein Datentyp festgelegt. Bei manchen Datentypen muss definiert werden, wie die Informationen aus den Quelldaten extrahiert werden können. Dies ist beispielsweise für einen Zeitstempel der Fall, bei dem zusätzlich angegeben werden muss, in welchem Format die Zeitinformation vorliegt, damit sie korrekt verarbeitet werden kann.

4.5 Funktionen der Konfigurationsoberfläche

Die Konfigurationsoberfläche enthält Funktionen, welche die Anforderungen zur *Einrichtung und Konfiguration (Unterabschnitt 3.3.1)* erfüllen. Die Funktionen werden auf drei Seiten aufgeteilt. Eine Übersichtsseite, eine Aktionen-Seite und die Konfigurationsseite. Für die Dokumentation der Konfigurationsoberfläche gibt es zusätzlich eine Hilfe-Seite.

Die Übersichtsseite zeigt, welche Rohdaten und transformierten Daten bereits verfügbar sind. Es ist angegeben, wie viele Datensätze vorliegen und wie aktuell die Datensätze sind. Abbildung 4.9 zeigt beispielhafte Werte auf der Übersichtsseite.

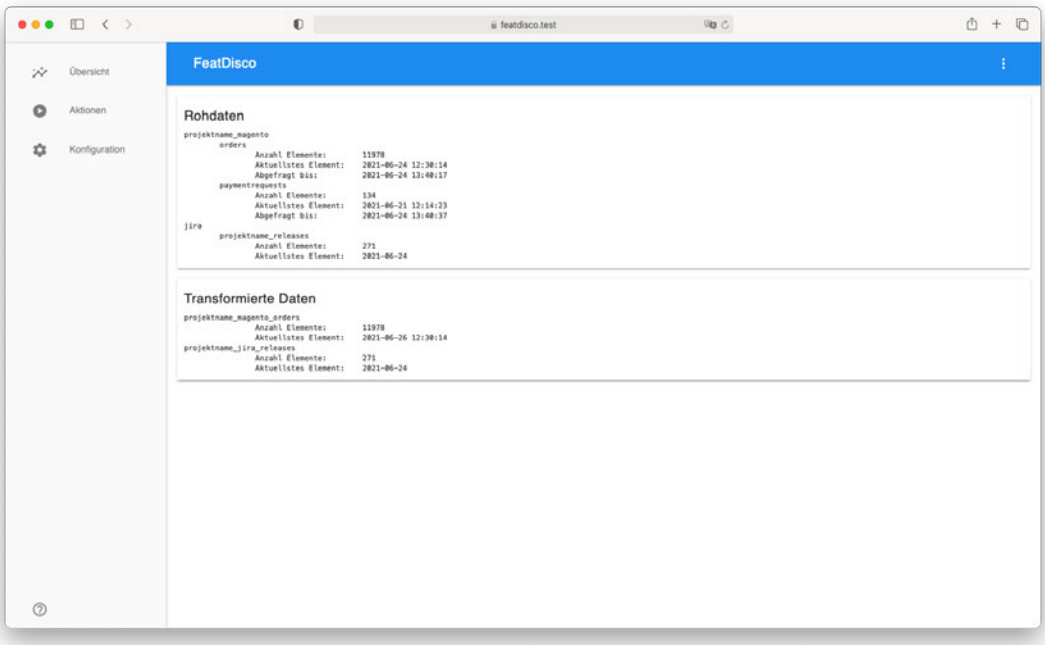


Abbildung 4.9: Übersichtsseite der Konfigurationsoberfläche

Die Aktionen-Seite ermöglicht das manuelle Starten von Aktionen. Es können die Rohdaten für bestimmte Abruf-Entitäten abgerufen und reindiziert werden. Letzteres ist bei der Veränderung mancher Konfigurationen notwendig. Dazu werden die Abruf-Entitäten ausgewählt und der Aktionsbutton geklickt. Außerdem können die Daten in das Format der Ziel-Entitäten transformiert und für die Visualisierungen geladen werden. Hierfür werden die Ziel-Entitäten ausgewählt, für die die Aktion durchgeführt werden soll, und der Aktionsbutton geklickt. Neben den Aktionsfeldern kann auf dieser Seite auch das Event-Log eingesehen werden, um zu überprüfen, ob Aktionen fehlerfrei ausgeführt wurden und wie lange ihre Ausführung gedauert hat. Abbildung 4.10 zeigt den Aufbau der Aktionen-Seite.

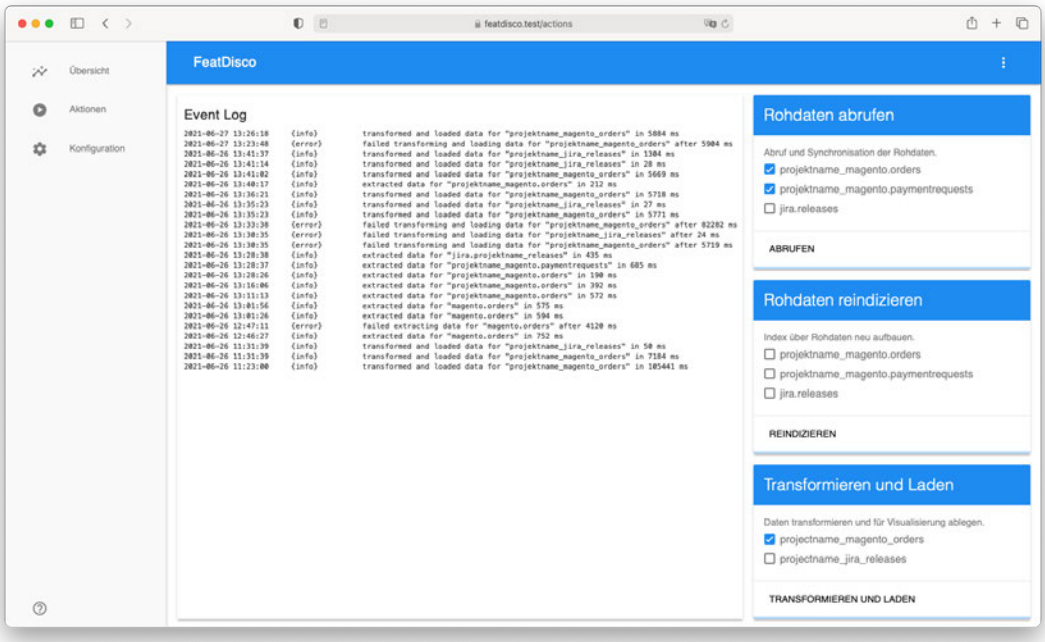


Abbildung 4.10: Aktionen-Seite der Konfigurationsoberfläche

Auf der Konfigurationsseite kann schließlich die Konfiguration neuer Abruf-Systeme und Ziel-Entitäten vorgenommen werden, wie es in *Unterabschnitt 4.3.2* und *Unterabschnitt 4.3.3* beschrieben ist. Abbildung 4.11 zeigt die Konfigurationsseite.

Benötigt die nutzende Person bei der Bedienung der Konfigurationsoberfläche Hilfe, findet sie diese auf der Hilfe-Seite. Die Hilfe-Seite führt durch die Einrichtungsschritte und dokumentiert die Bedeutung aller Konfigurationsfelder für die verschiedenen Abruf-Systeme, sowie die Felder und verfügbaren Funktionen zur Transformation der Daten. Abbildung 4.12 zeigt einen Ausschnitt der Hilfe-Seite.

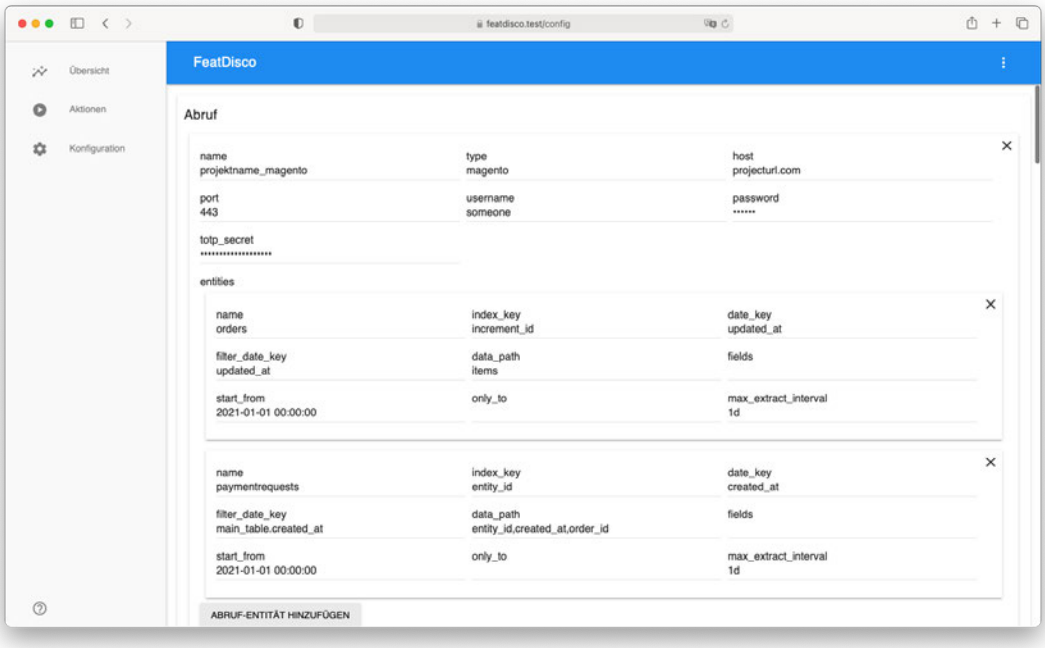


Abbildung 4.11: Konfigurationsseite der Konfigurationsoberfläche

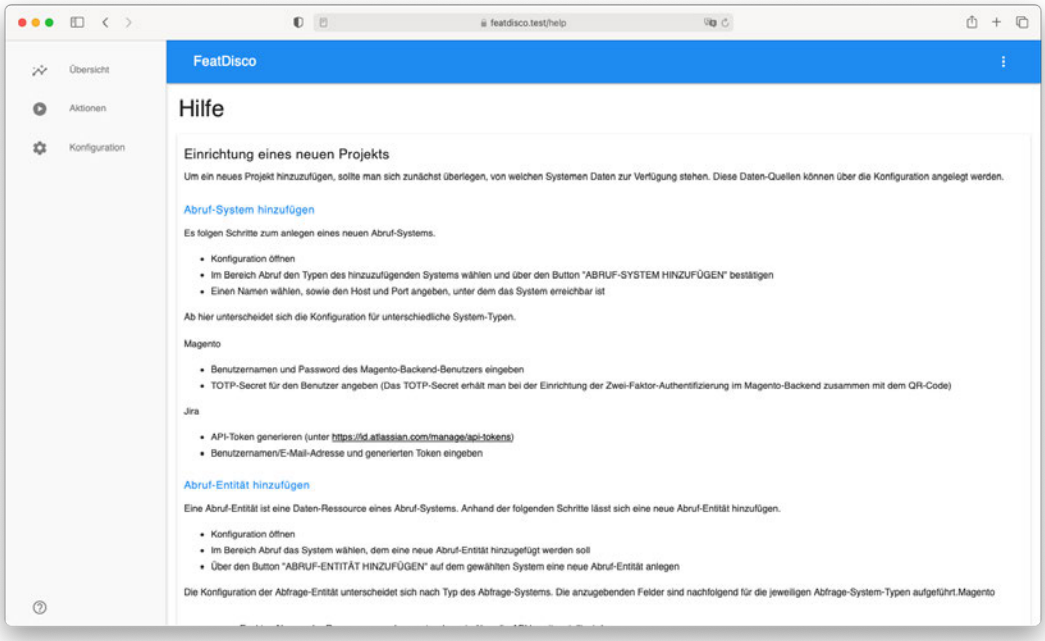


Abbildung 4.12: Hilfe-Seite der Konfigurationsoberfläche

4.6 Hochfahren des Systems

Da alle Komponenten des Systems in Docker Containern ausgeführt werden, gestaltet sich das Deployment einfach. Es kann somit innerhalb kurzer Zeit auf jedem Server aufgesetzt werden, auf dem Docker und Docker Compose installiert sind. Alle Abhängigkeiten werden beim Bauen der Images installiert und ein Netzwerk erstellt, über das die Services untereinander kommunizieren können.

Damit die Daten über die Lebensdauer der Docker Container hinaus erhalten bleiben und nach einem Neustart des Systems nicht verloren gehen, werden bestimmte Verzeichnisse in den Containern als Volumes gemountet. Dies geschieht bereits automatisch beim Starten anhand der Docker Compose Konfiguration. Zum Hochfahren muss lediglich der Befehl `docker-compose up -d` ausgeführt werden. Das System ist dann einsatzbereit.

4.7 Dokumentation der Anwendung

Die Bedienung der Anwendung soll möglichst intuitiv sein. Besonders bei den ersten Schritten der Konfiguration soll die oder der Anwender*in aber unterstützt werden. Die Dokumentation für die Konfiguration befindet sich direkt auf der Konfigurationsoberfläche und ist über einen Eintrag im seitlichen Menü erreichbar. Einen Ausschnitt der Hilfe-Seite zeigt Abbildung 4.12.

Wichtige Informationen zur Infrastruktur sowie zum Starten und Stoppen der Anwendung sind in der README.md Datei des Projektes dokumentiert.

5 Implementierung

Dieses Kapitel enthält wichtige Details zur Implementierung des zuvor entworfenen Systems und zur Durchführung der Auswertungen.

Zunächst wird die Implementierung zum Abruf und zur Vorbereitung der Daten erläutert. Dazu werden die Aufgaben der einzelnen Pakete von *FeatDisco* beschrieben und zwei Algorithmen erklärt, die für den Abruf aus dem Data Lake und für das Transformieren der Daten entwickelt wurden.

Anschließend wird die Umsetzung der Infrastruktur und das Hosting des Systems geschildert.

Im nächsten Schritt wird erklärt, wie die Visualisierungen auf dem Dashboard erstellt werden. Dabei wird die Umsetzung konzipierter Visualisierungen beschrieben.

5.1 Abruf und Vorbereitung der Daten

FeatDisco ist in JavaScript implementiert. Die serverseitige JavaScript-Umgebung Node.js ermöglicht, dass die Geschäftslogik ebenfalls in JavaScript implementiert ist. JavaScript ist in der Webentwicklung für die Entwicklung der Benutzeroberfläche unerlässlich. Daher bietet es sich an, die Sprache einheitlich auch für die serverseitige Geschäftslogik einzusetzen. Die Architektur von Node.js ermöglicht die Verwendung einer ausdrucksstarken und funktionalen Sprache für die Serverprogrammierung ohne Leistungseinbußen (Tilkov und Vinoski, 2010, S. 83). Die Installation vieler nützlicher Bibliotheken und Abhängigkeiten ist dank des Paket-Managers npm unkompliziert (Tilkov und Vinoski, 2010, S. 83).

Asynchrone Programmierung lässt sich mit JavaScript besonders gut aufgrund der ereignisgesteuerten Architektur umsetzen. In vielen Fällen müssen folgende Operationen nicht auf den Abschluss der vorherigen Funktion warten und können parallel laufen. Ist

die vorherige Funktion abgeschlossen, kann eine Callback-Funktion aufgerufen werden, wenn dessen Arbeitsschritte von den vorherigen abhängig sind.

Für das Starten des Web-Servers wird das Webframework Express.js verwendet. Es wird mit dem Paket-Manager npm im Node.js-Projekt installiert. Die Routen, welche aufgerufen werden können, werden in Express.js registriert und der Server auf einem festgelegten Port gestartet. Für die Routen werden callback-Funktionen implementiert, die aufgerufen werden, sobald die Route angefragt wird. Die callback-Funktionen verwenden Funktionen aus den nachfolgend beschriebenen Paketen.

Die Webseiten der Konfigurationsoberfläche werden mit der Template-Engine Handlebars.js gerendert. Handlebars.js wird ebenfalls mit npm installiert und als view engine für Express.js registriert. Es gibt layouts, partials und views, wobei layouts den Grundaufbau einer Seite enthalten und partials einzelne Seitenkomponenten. Die views enthalten den spezifischen Inhalt einer Seite und können mit einem bestimmten layout gerendert werden. Die partials können in den views und layouts an einer bestimmten Stelle eingebunden werden.

5.1.1 Pakete von FeatDisco

auth

Das „auth“-Paket ist für die Authentifizierung der Benutzerin oder des Benutzers zuständig, die oder der eine Anfrage stellt. Beim Aufrufen von *FeatDisco* werden ein Benutzername und Passwort abgefragt. Nach der Eingabe werden die Anmeldedaten als POST-Request an die „/login“-Route gesendet. Die login Methode prüft, ob die oder der Benutzer*in berechtigt ist und das Passwort stimmt. Als zusätzlichen Schutzmechanismus hat ein*e Benutzer*in nur eine begrenzte Anzahl an Anmeldeversuchen. Ist die oder der Benutzer*in berechtigt, wird eine Session angelegt und ein generierter Token zurückgesendet. Der Token wird als Cookie gespeichert, sodass die oder der Benutzer*in am System angemeldet bleibt. Nach einer gewissen Zeit läuft die Session ab und die oder der Benutzer*in muss sich erneut anmelden. Die Prüfung, ob eine Anfrage berechtigt ist, übernimmt eine Middleware. Diese wird für alle zu schützenden Routen in Express.js eingebunden und vor der callback-Funktion der Route ausgeführt.

config

Dieses Paket ist für die Anzeige der Konfigurationsseite, Verarbeitung und Speicherung der Konfiguration nach Änderungen über die Konfigurationsseite und Bereitstellung der Konfiguration für die Ausführung von Aktionen verantwortlich. Beim Aufruf der Konfigurationsseite werden aus der im JSON-Format gespeicherten Konfiguration Formularfelder generiert. Die verfügbaren Abruf-System-Typen werden ermittelt und für die Konfiguration bereitgestellt. Zur Ermittlung der verfügbaren Abruf-System-Typen wird das Plug-In-Verzeichnis nach Systemen durchsucht. Dabei werden auch die zu konfigurierenden Felder für diese System-Typen aus den „config.json“-Dateien der Plug-Ins geladen. Beim Hinzufügen neuer Abruf-Systeme werden Formularfelder entsprechend dieser Konfiguration generiert. Beim Speichern der Konfiguration werden die Formulardaten wieder in das JSON-Format umgewandelt.

extract

Das „extract“-Paket ist verantwortlich für den Abruf und die Speicherung der Rohdaten. Um den Abruf zu starten, werden die Namen der abzurufenden Entitäten als POST-Request an die „/extract“-Route gesendet. Nun wird die Implementierung für das abzurufende System angefordert und dynamisch geladen. Ist diese vorhanden, wird der Abruf anhand dieser Implementierung gestartet. Die spezifische Abruf-Implementierung erweitert eine Basisimplementierung. Es müssen nur Funktionen implementiert werden, die besonders für das Abruf-System sind. Die Basisfunktionen des Abrufs werden nicht erneut implementiert. Sie enthalten vielmehr eine einheitliche Funktion für das Speichern der Daten und aktualisieren des Indexes.

Für den Abruf wird der Zeitstempel des aktuellsten Datensatzes aus dem Index geladen. Ist kein maximales Abruf-Intervall angegeben, werden die Daten bis zum Zeitpunkt des Abrufs angefragt, andernfalls für die Länge des erlaubten Abruf-Intervalls. Wenn ein Zeitstempel konfiguriert wurde, bis zu dem abgerufen werden soll, wird dieser stattdessen verwendet. Die Daten werden für den festgelegten Zeitraum abgerufen. Wenn der Abruf erfolgreich war, werden sie anhand der Basismethode gespeichert und der Index aktualisiert. Wenn der Abruf nicht erfolgreich war, kann es daran liegen, dass der API-Token abgelaufen ist. In diesem Fall wird ein neuer Token abgerufen und der Abruf der Daten erneut versucht.

Für das manuelle Starten des Abrufs über die Konfigurationsoberfläche werden die verfügbaren Abruf-Systeme aus der Konfiguration geladen. Diese können von der benutzenden Person ausgewählt werden, um den Abruf für diese Systeme zu starten. Auf die gleiche Weise kann der Neuaufbau des Indexes veranlasst werden.

Die auf der Übersichtsseite angezeigten Informationen zu den abgerufenen Daten werden aus den Indizes geladen.

helper

Das „helper“-Paket enthält Hilfsfunktionen, die in anderen Paketen benötigt werden. Es gibt Funktionen, um bestimmte Darstellungen für einen Zeitstempel zu erzeugen oder einen Zeitstempel für die im System konfigurierte Zeitzone umzuwandeln. Weitere Hilfsfunktionen prüfen, ob mehrere Verzeichnisse existieren und erstellen sie gegebenenfalls oder stoppen die Zeit für die Ausführung einer Aktion.

log

Dieses Paket bietet eine Schnittstelle für das Loggen von Ereignissen. Es können ein Typ, zum Beispiel „error“ oder „info“, und eine Nachricht übergeben werden. Das Logging blockiert die aufrufende Methode nicht, sondern läuft asynchron, sodass die Ausführung nicht verlangsamt wird. Die Nachricht wird mit einem Zeitstempel geloggt. Die Event-Logs werden auf der Aktionen-Seite angezeigt.

transformLoad

Dieses Paket ist verantwortlich für die Transformierung der Daten und das Laden der Daten in einen Speicher, der für die Visualisierungen eingebunden werden kann. Dieser Prozess wird durch einen POST-Request an die „/transformLoad“-Route gestartet, wobei die zu transformierenden Ziel-Entitäten übergeben werden. Für die Entität wird überprüft, welches das aktuellste Element ist, das bereits verarbeitet wurde. Nur neuere Elemente werden aus dem Data Lake abgerufen (vgl. *5.1.2 Algorithmus zum Abruf aus dem Data Lake*). Die maximale Anzahl an Elementen, die gleichzeitig verarbeitet werden,

ist beschränkt, um nicht zu viel Arbeitsspeicher zu belegen. Wenn der erste Batch an Daten transformiert und gespeichert wurde, wird der Prozess mit dem nächsten gestartet, bis alle Daten transformiert wurden.

Nachdem die Daten transformiert wurden (vgl. *5.1.3 Algorithmus zum Transformieren der Daten*), werden sie in Elasticsearch gespeichert. Dazu werden die Eigenschaften der Felder aus der Konfiguration abgerufen. Dies betrifft den Datentyp und gegebenenfalls das Format, in dem Daten vorliegen. Es wird geprüft, ob der Index in Elasticsearch bereits existiert. Wenn der Index nicht existiert, wird dieser erstellt und das Mapping für den Index in Elasticsearch anhand der Eigenschaften der Felder festgelegt (vgl. *2.4.3 Mapping*). Im nächsten Schritt werden alle transformierten Daten in Elasticsearch gespeichert. Abschließend wird die neue Anzahl an Dokumenten im Elasticsearch-Index abgefragt und für die Anzeige auf der Übersichtsseite gespeichert. Alle Interaktionen mit Elasticsearch finden über die REST-API statt (vgl. *2.4.4 REST-API*).

Die Transformation kann auf der Konfigurationsoberfläche manuell gestartet werden. Die verfügbaren Ziel-Entitäten werden aus der Konfiguration geladen und angezeigt. Die zu transformierenden Ziel-Entitäten können ausgewählt und die Aktion gestartet werden.

Die Übersichtsseite zeigt die Aktualität der transformierten Daten und die Anzahl transformierter Dokumente pro Ziel-Entität.

5.1.2 Algorithmus zum Abruf aus dem Data Lake

In Abbildung 5.1 wird der Algorithmus zum Abruf aus dem Data Lake in vereinfachter Form dargestellt. Es ist ein Mechanismus implementiert, der es ermöglicht, nur eine bestimmte Anzahl Elemente abzurufen und nach Verarbeitung der Elemente die nächsten Elemente abzurufen. Um die Übersicht zu bewahren wurde dieser Mechanismus in der Abbildung nicht berücksichtigt.

Der Algorithmus erhält als Eingabeparameter die Namen des Systems und der Entität und einen Zeitstempel, der festlegt, ab welchem Datensatz die Daten geladen werden. Der Index wird anhand des Systems und der Entität geladen. Eine Variable zum Sammeln der Daten wird initialisiert. Für alle Dateien aus dem Index werden die Schlüssel der Entitäten ermittelt, dessen neuste Version in der aktuellen Datei enthalten sind. Die aktuelle Datei wird eingelesen und für jeden der ermittelten Schlüssel wird die Entität aus den Daten extrahiert. Ist der Zeitstempel der Entität später als der Zeitstempel aus

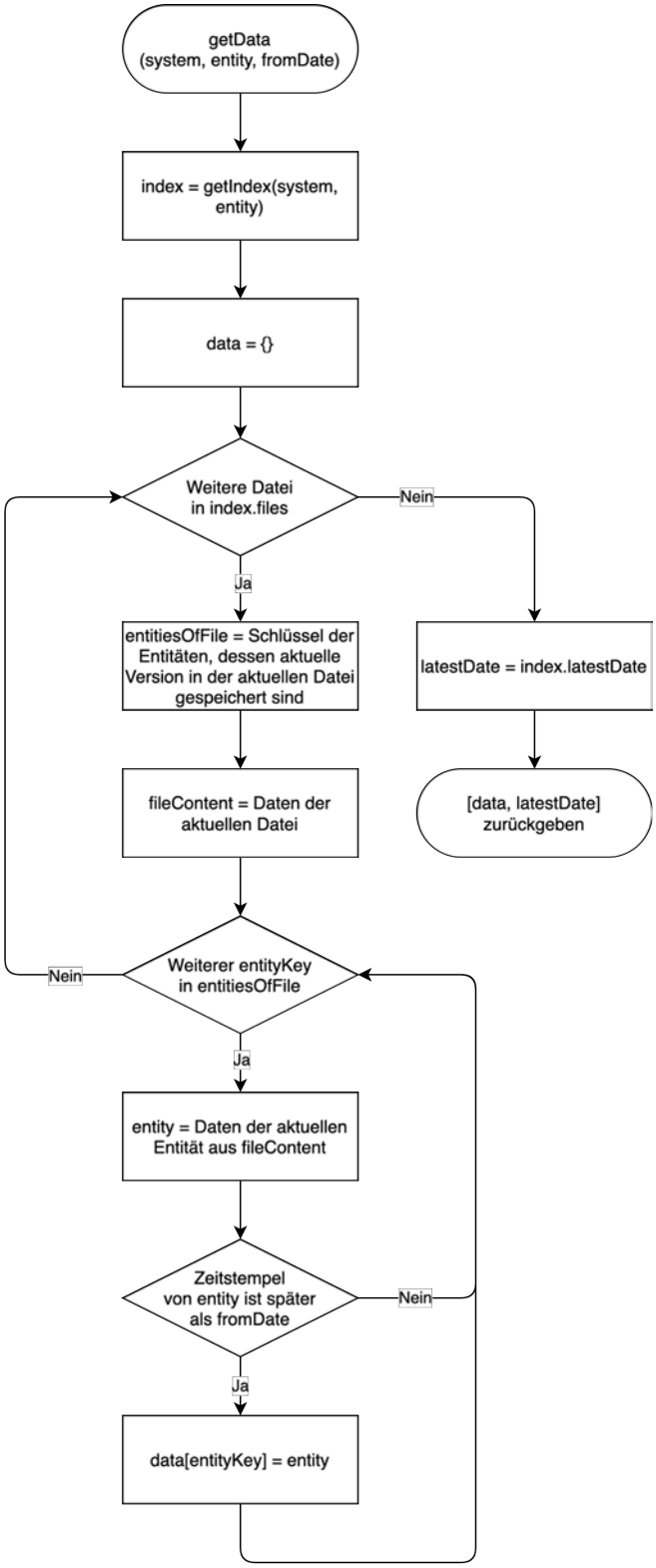


Abbildung 5.1: Ablaufdiagramm des Algorithmus zum Abruf aus dem Data Lake

den Eingabeparametern, wird die Entität für die Rückgabe gesammelt. Nachdem alle Dateien gelesen wurden, werden die Daten zusammen mit dem aktuellsten Zeitstempel der Daten zurückgegeben.

5.1.3 Algorithmus zum Transformieren der Daten

Abbildung 5.2 zeigt in vereinfachter Form den Algorithmus zum Transformieren der Daten. Der Algorithmus wird mit den Rohdaten und der Konfiguration für die Transformation gestartet. Eine Variable zum Sammeln der transformierten Daten wird initialisiert. Als Haupt-Entität wird die erste angegebene Quell-Entität verwendet. Für jedes Dokument in den Rohdaten der Haupt-Entität wird jedes Ziel-Feld aus der Konfiguration für die Transformation verarbeitet. Zum besseren Verständnis kann das Konfigurationsbeispiel aus Abbildung 4.6 herangezogen werden.

Ist die Funktion des Ziel-Feldes „copy“ und die Feld-Quelle entspricht der Haupt-Entität, wird der Feld-Wert aus dem aktuellen Dokument in die Ziel-Entität kopiert. Ist als Feld-Quelle eine andere Quell-Entität angegeben, muss diese aus dem Data Lake abgerufen werden. Der Feld-Wert aus dem abgerufenen Dokument wird dann in die Ziel-Entität kopiert.

Für die Funktion „exists“ wird das Dokument, bei dem der Wert des für die zweite Feld-Quelle angegebenen Feldes, dem Wert des Feldes aus der ersten Feld-Quelle entspricht, aus dem Data Lake geladen. Als Wert der Ziel-Entität wird ein Wahrheitswert festgelegt, abhängig davon, ob im vorherigen Schritt ein Dokument gefunden wurde.

5.2 Infrastruktur und Hosting

In diesem Abschnitt wird beschrieben, wie das System bereitgestellt wird.

Die Bestandteile des Systems laufen in Docker Containern, die mit Docker Compose erstellt und gestartet werden. Mit welchen Einstellungen die Container gestartet werden, ist in der `docker-compose.yml` festgelegt. Quellcode 5.1 zeigt einen Ausschnitt aus der `docker-compose.yml`.

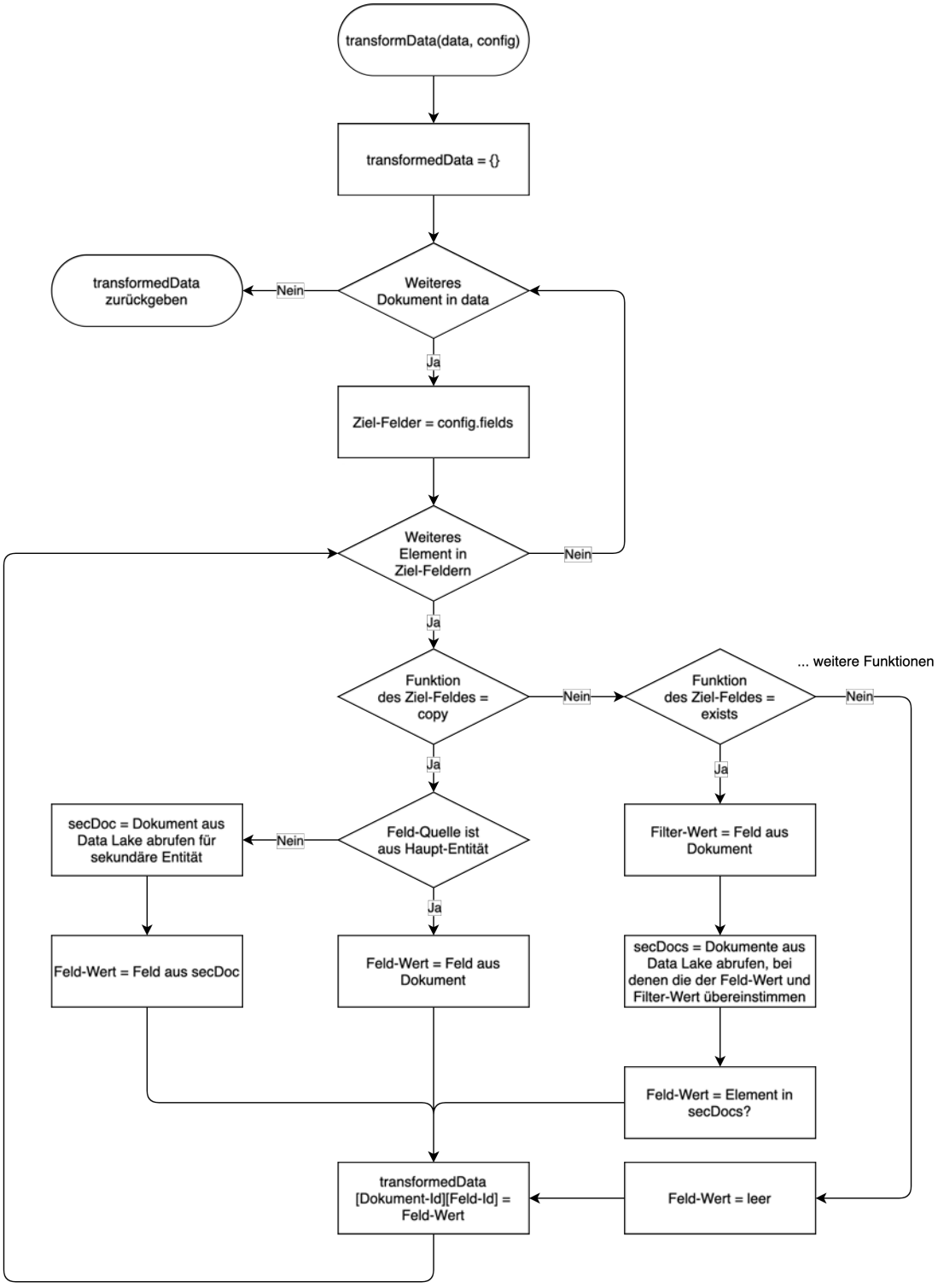


Abbildung 5.2: Ablaufdiagramm des Algorithmus zum Transformieren der Daten

```
version: '3'
services:
  nginx:
    image: nginx:1.21.0
    container_name: nginx
    ports:
      - 80:80
      - 443:443
    volumes:
      - ./nginx:/etc/nginx/conf.d
      - ./certs:/etc/certs
    depends_on:
      - featdisco
      - grafana
    restart: unless-stopped
  featdisco:
    build: .
    container_name: featdisco
    ports:
      - 4040:4040
    volumes:
      - ./config:/app/config
      - ./data:/app/data
      - ./log:/app/log
    depends_on:
      - elasticsearch
      - grafana
    restart: unless-stopped
```

Quellcode 5.1: Ausschnitt aus der Docker Compose Konfiguration

Der Proxyserver läuft in einem Container, der aus dem „nginx“-Image erstellt wird. Die internen Ports 80 und 443 werden nach außen auf den gleichen Ports freigegeben. Die Server-Konfiguration und die Server-Zertifikate werden als Volumes gemountet. Das Server-Zertifikat wurde selbst ausgestellt, für die IP unter der das System erreichbar ist. Sobald das System über einen Domain-Namen verfügbar ist, können die Zertifikate auto-

matisch mit Certbot erstellt werden. Certbot ruft vor Ablauf eines Zertifikates ein neues ab, welches von einer vertrauenswürdigen Zertifizierungsstelle ausgestellt wird.

Wenn der Proxyserver gestartet wird, sollen auch *FeatDisco* und Grafana gestartet sein, da der Proxyserver Ressourcen dieser Services weiterleitet. Wird der Container unerwartet gestoppt, soll er automatisch neu gestartet werden, sodass die Seite verfügbar bleibt. Dies geschieht auch, wenn das System neu gestartet wird.

Für *FeatDisco* wird ein weiterer Container erstellt. Das Image für diesen Container wird anhand der `Dockerfile` im Projektverzeichnis erzeugt. Der Port 4040 wird freigegeben. Auf diesem Port läuft der Express.js-Server. Die Verzeichnisse für die Konfiguration, die Rohdaten, sowie Logdateien werden als Volumes eingebunden, sodass die Daten auch nach einem Neustart des Containers verfügbar bleiben. *FeatDisco* interagiert mit Elasticsearch und Grafana, daher sollen die Container dieser Anwendungen beim Start von FeatDisco gestartet sein. *FeatDisco* soll bei unerwartetem Stoppen des Containers ebenfalls neu gestartet werden.

Die Container für Elasticsearch und Grafana werden analog zu nginx konfiguriert. Die gemounteten Verzeichnisse müssen den Benutzern gehören, welche die Anwendung starten, damit der Zugriff möglich ist. Am Beispiel von Elasticsearch und Grafana sind die Benutzer 1000 und 472.

Docker Compose erstellt ein Netzwerk, über das die Services miteinander kommunizieren können. Ein Service ist über seinen Container Namen als Hostname erreichbar. Somit kann sich beispielsweise Grafana mit Elasticsearch über die Adresse http://elasticsearch:9200 verbinden.

Zur Bereitstellung der Anwendung wird ein Server bei einem Cloud-Anbieter gemietet. Der Server stellt 4 GB Arbeitsspeicher und 2 CPUs zur Verfügung. Bei Tests mit weniger Arbeitsspeicher traten Fehler im Zusammenhang mit Elasticsearch auf, da nicht genügend Speicher angefragt werden konnte.

Für den Server wird eine Firewall eingerichtet. Sie erlaubt den Zugang von außen ausschließlich über die Ports, auf denen nginx die Anwendungen über eine sichere Verbindung bereitstellt, und über SSH für den Zugriff über die Shell.

Über SSH können auf dem Server die Befehle zum Einrichten des Systems ausgeführt werden. Das Repository, welches den Code für die Anwendung enthält, wird mit dem

Versionskontrollsystem Git auf dem Server heruntergeladen. Anschließend wird das System mit Docker Compose gestartet. Dabei werden die benötigten Images heruntergeladen und im Fall von *FeatDisco* aus dem Quellcode erzeugt. Nachdem Docker Compose den Start der Container bestätigt hat, ist das System verfügbar.

5.3 Erstellung von Visualisierungen

In diesem Abschnitt wird die Umsetzung der Anforderungen zur Analyse eines Onlineshops beschrieben (vgl. *3.4 Analyse eines Onlineshops*).

5.3.1 Auswertung von Zahlungsarten

Um eine erste Feature-Auswertung zu testen, wurden Analysen zur Nutzung der verfügbaren Zahlungsarten erstellt.

Nach dem GQM-Modell (vgl. *2.6.1 Auswahl der Daten*) wird das Ziel, die Fragestellungen und die Maßnahmen zur Messung ermittelt.

Das Ziel ist:

Analyse der Zahlungsarten eines Onlineshops in Hinblick auf die Nutzungshäufigkeit und die Umsatzhöhe zur Evaluierung des Mehrwertes.

Fragestellungen sind:

- Wie viele Bestellungen werden pro Zahlungsart aufgegeben?
- Wie hoch ist die Umsatzsumme der Bestellungen pro Zahlungsart?
- Wie hoch ist die durchschnittliche Bestellsumme pro Zahlungsart?

Maßnahmen zur Messung sind:

- Zählen aller Bestellungen gruppiert nach Zahlungsart.
- Summieren der Gesamtsummen aller Bestellungen pro Zahlungsart.
- Durchschnitt der Gesamtsummen aller Bestellungen pro Zahlungsart berechnen.

Für die Messungen werden die Bestelldaten benötigt. Diese enthalten die Zahlungsart und die Gesamtsumme einer Bestellung.

Für die Visualisierung wird ein Dashboard für das Pull-Szenario erstellt (vgl. *2.6.2 Wahl der Visualisierungstechnik*), da die nutzende Person die Informationen selbst abruft und ihre oder seine Aufmerksamkeit nicht aufgrund von bestimmten Ereignissen erregt werden muss. Um sich die Auswertung für verschiedene Zeiträume anzeigen zu lassen, bedient sie oder er das Dashboard.

Die Wahl des Diagrammtyps fällt auf ein Tortendiagramm, da hier am besten das Verhältnis zwischen den Zahlungsarten abgelesen werden kann.

Beispielhafte Visualisierungen zum oben genannten Ziel zeigen die Abbildungen 5.3, 5.4 und 5.5.

Abbildung 5.3 zeigt die Anzahl an Bestellungen gruppiert nach der Zahlungsart. Die Visualisierung veranschaulicht die Nutzungshäufigkeit der einzelnen Zahlungsarten.

Abbildung 5.4 zeigt die durchschnittliche Bestellsumme pro Zahlungsart. Diese Visualisierung zeigt, mit welchen Zahlungsarten die teureren Bestellungen getätigt werden. Zahlungsarten die seltener genutzt werden, aber bei höheren Bestellsummen bevorzugt werden, treten hier in den Vordergrund.

Abbildung 5.5 zeigt den Umsatz pro Zahlungsart. Diese Auswertung vereint die beiden vorhergehenden Auswertungen und gibt einen Gesamteindruck über die Mehrwerte der Zahlungsarten.

Abbildung 5.3: Visualisierung der Anzahl an Bestellungen pro Zahlungsart

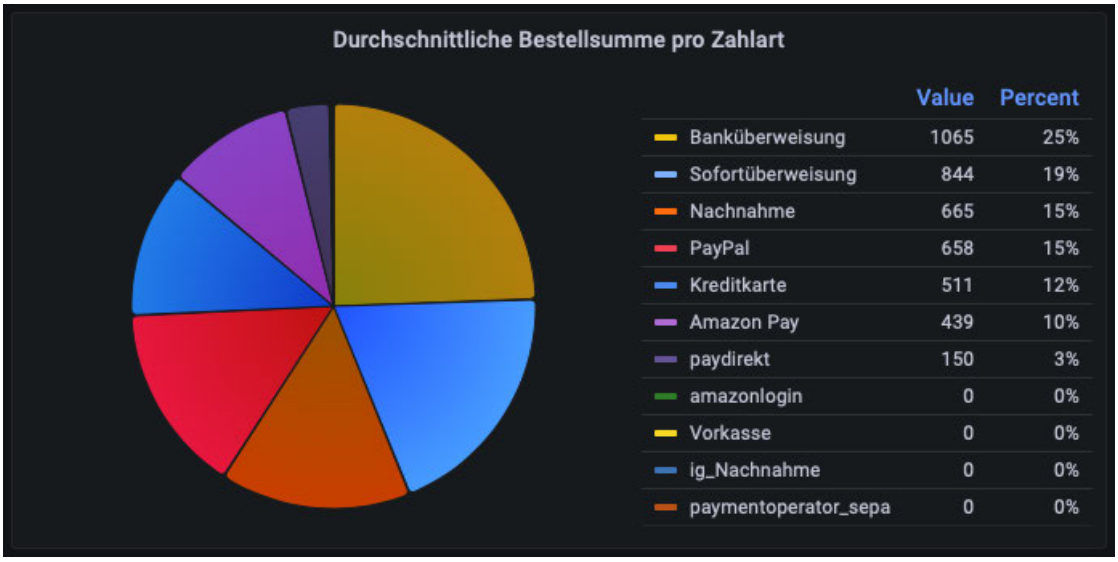


Abbildung 5.4: Visualisierung der durchschnittlichen Bestellsumme pro Zahlungsart

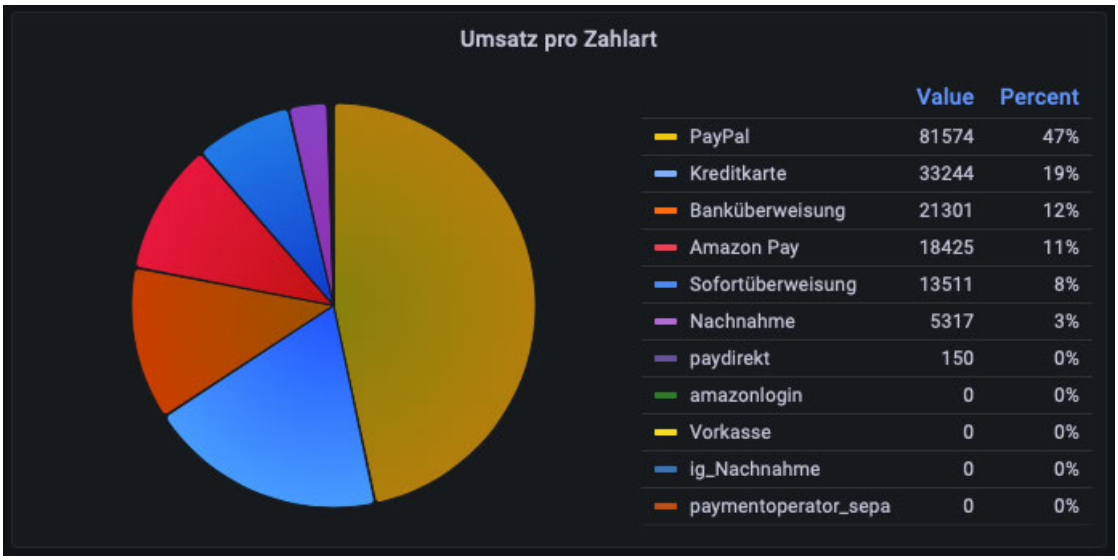


Abbildung 5.5: Visualisierung des Umsatzes pro Zahlungsart

5.3.2 Darstellung der Umsätze in Zusammenhang mit den Releases

Das Ziel dieser Visualisierung ist die Analyse des Umsatzverlaufs eines Onlineshops in Zusammenhang mit den Releasezeitpunkten zur Einordnung von Umsatzveränderungen.

Die Fragestellung ist: Wie verändert sich der Umsatz in Abhängigkeit zum Release bestimmter Features?

Die Maßnahme zur Messung ist: Markieren der Releasezeitpunkte im Umsatzverlauf.

Um den Umsatz in Zusammenhang mit den Releases darzustellen, wird der Umsatzverlauf visualisiert und Events im Verlauf angezeigt. Die Events werden als Anmerkungen an den Stellen auf der Zeitachse angezeigt, wo ein neues Release erstellt wurde. In Grafana können Anmerkungen aus Datenquellen erzeugt werden. Es wird eine Datenquelle erstellt, welche Zeitinformationen, Beschreibung der Releases und Tag des Releases enthält. Wird der Mauszeiger über die Anmerkung im Diagramm geführt, werden die zusätzlichen Informationen angezeigt. Es kann so der Verlauf des Umsatzes vor und nach einem Event bewertet werden.

Für die Visualisierung werden die Bestelldaten benötigt, sowie Releaseinformationen. Um den Verlauf der Umsätze darzustellen wird als Visualisierung eine Zeitreihe verwendet.

Abbildung 5.6 Zeigt den Umsatzverlauf mit Markierungen an den Releasezeitpunkten. Die Visualisierung zeigt neben dem Umsatz auch die Anzahl an Bestellungen und bestellten Artikeln. Für die Releases kann der Release-Tag und die Release-Beschreibung angezeigt werden. Wurden entscheidende Features veröffentlicht, ist bestenfalls eine Steigerung des Umsatzes erkennbar.

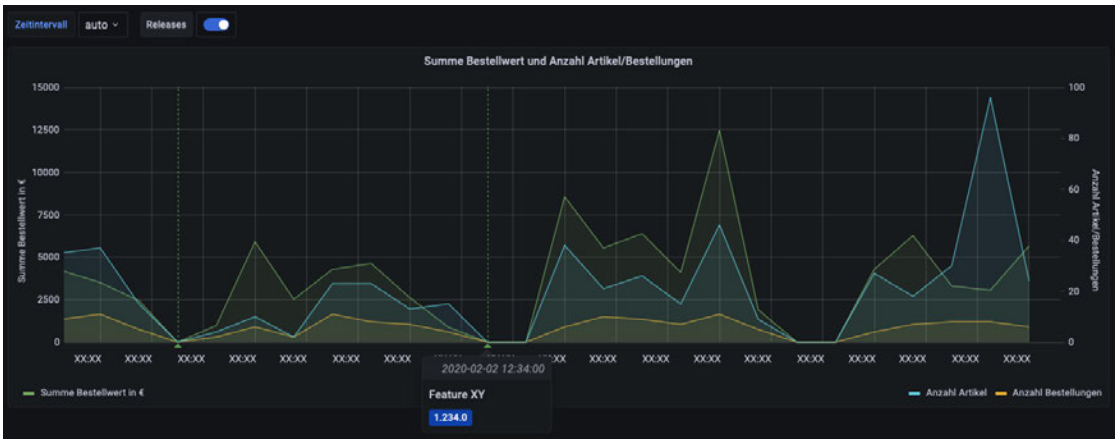


Abbildung 5.6: Visualisierung des Umsatzverlaufs mit Releasezeitpunkten

5.3.3 Filtern des Umsatzverlaufs durch Zusammenhang zu einem Feature

Eine weitere Möglichkeit, um ein Feature auszuwerten, ist es den Zusammenhang eines Bestellabschlusses mit der vorherigen Nutzung eines Features herzustellen. Dazu kann bei der Daten-Transformation ein Feld in der Ziel-Entität angelegt werden, welches einen Wahrheitswert enthält, ob die Bestellung in Zusammenhang mit der Nutzung des Features steht.

Abbildung 5.7 zeigt den Umsatzverlauf für Bestellungen, die in Zusammenhang mit einem neu implementierten Feature stehen. Das Diagramm zeigt auch die Anzahl an Bestellungen und bestellten Artikeln. Es ist zu erkennen, wie die Häufigkeit von Bestellungen und damit der Umsatz in Zusammenhang mit dem Feature zunehmen.

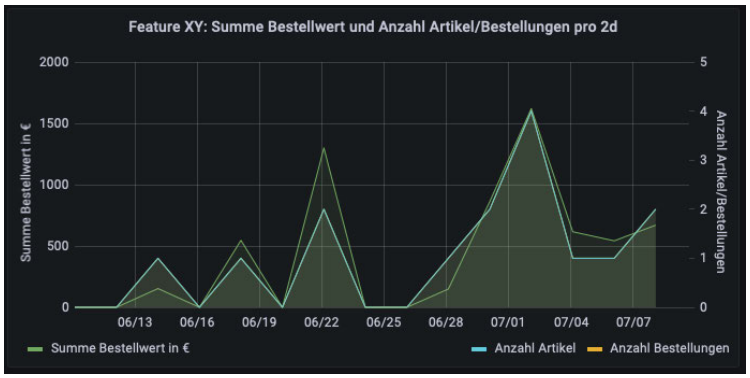


Abbildung 5.7: Visualisierung des Umsatzverlaufs gefiltert nach Zusammenhang mit Feature

6 Evaluation

Um eine erste Einschätzung zu erhalten, wie die Konfigurationsoberfläche und die Visualisierungen auf dem Dashboard von potenziellen Nutzer*innen des Systems wahrgenommen werden, wurde das System für eine Evaluationsphase bereitgestellt. Am Ende der Evaluationsphase wurden die Teilnehmer*innen anhand eines Fragebogens befragt. Der Fragebogen ist im *Anhang* beigefügt.

Im Folgenden wird beschrieben, wie das System für die Evaluation bereitgestellt wurde. Anschließend werden die Antworten der Teilnehmer*innen ausgewertet.

6.1 Bereitstellung

Das System wurde auf einem Server installiert (vgl. *5.2 Infrastruktur und Hosting*). Den Testpersonen wurden die benötigten Anmeldedaten übermittelt. Sie haben die Aufgabe erhalten, zwei festgelegte Abruf-Systeme und zwei Ziel-Entitäten zu konfigurieren. Dazu wurden die benötigten Informationen bereitgestellt. Anschließend sollten sie den Abruf und die Transformation starten und das Ergebnis auf dem Dashboard betrachten.

6.2 Auswertung

An der Evaluation haben sechs Entwickler*innen aus zwei Entwicklungsteams der Tudock GmbH teilgenommen. Nachdem sie Erfahrungen mit dem System gesammelt haben, haben sie den Fragebogen beantwortet (siehe *Anhang*). Zu Beginn wurden Fragen zur Motivation, das System zu nutzen, gestellt. Es folgten Fragen zur Konfiguration der Abruf-Systeme und der Transformation, zur Übersichtsseite und zur Ansicht des Dashboards. Abschließend wurde der Gesamteindruck abgefragt. Wenn es nicht anders angegeben wurde, haben alle sechs Teilnehmer*innen eine Antwort abgegeben.

Motivation

Anhand der Fragen zur Motivation soll festgestellt werden, wie groß die Bereitschaft ist, das System zur Auswertung von implementierten Features einzusetzen.

Die Fragen 1 und 2 geben einen Eindruck, wie viele Features monatlich ausgewertet werden können. Der überwiegende Anteil der Teilnehmer*innen hat angegeben an 2–3 Features pro Monat zu arbeiten. Eine Antwort liegt knapp darunter eine knapp darüber. Die Hälfte der Teilnehmer*innen hat angegeben, dass die Auswertung für 20–50 % der Features nützlich ist. Zwei Antworten lagen höher, eine etwas niedriger. Anhand dieser Antworten ist zu erwarten, dass pro Monat mehrere Features implementiert werden, für die eine Auswertung nützlich ist.

Die Fragen 3–6 ermitteln, wie hoch das Interesse an der Nutzung eines Systems zur Auswertung von implementierten Features ist und welche Auswertungen und Einsatzgebiete bevorzugt werden. Alle Teilnehmer*innen finden es interessant, mehr Informationen zum Mehrwert bereitgestellter Features zu erhalten, die Hälfte von ihnen sogar sehr interessant. Insbesondere werden als Auswertung die Nutzungsfrequenz von Features, die Konversionsrate in Zusammenhang mit Features, der Anteil an Kaufabbrüchen sowie die Nutzungsdauer eines Features bis zum Kaufabschluss gewünscht. Die Konversionsrate beschreibt in diesem Zusammenhang das Verhältnis von allen Personen, die den Shop aufgerufen haben oder ein Feature genutzt haben zu den Personen, die auch zu Käufer*innen wurden, also einen Kauf abgeschlossen haben. Bei den Fragen zu den Einsatzgebieten waren die Teilnehmer*innen geteilter Meinung. Ein Drittel standen der Frage zur Motivation durch Betrachten des Dashboards in Teamsitzungen neutral gegenüber. Die übrigen Teilnehmer*innen würden es als motivierend bis sehr motivierend empfinden. Zwei Drittel der Teilnehmer*innen würde es motivieren, wenn das Dashboard im Büro platziert werden würde. Eine*r der sechs Teilnehmer*innen gab an, dass es ihn sehr motivieren würde. Ein weiterer gab hingegen an, dass es ihn eher nicht motivieren würde.

Die Antworten zeigen, dass durchaus ein Interesse für die Auswertung von Features besteht, bei einem Großteil der Befragten sogar ein hohes Interesse. Dass die Teilnehmer*innen angegeben haben, welche Auswertungen sie interessieren würden, verstärkt den Eindruck, dass Interesse besteht. Zudem helfen die Antworten bei der Verbesserung des Systems. Da ein Großteil der Teilnehmer*innen die Einsatzgebiete motivierend finden würde, ist es durchaus aussichtsreich die verschiedenen Einsatzgebiete auszuprobieren.

Die Fragen 7–9 schlüsseln auf, ob die Teilnehmer*innen dazu bereit wären, das System regelmäßig zu pflegen und in welcher Frequenz sie Informationen abrufen würden. Ein Drittel der Teilnehmer*innen sind sich nicht sicher, ob sie das System pflegen würden. Allerdings hat kein*e Teilnehmer*in angegeben, dass sie oder er es definitiv nicht oder eher nicht pflegen würde. Die übrigen Teilnehmer*innen würden es pflegen, ein Drittel der Befragten sind sich sogar sicher dieses zu tun. Die Hälfte der sechs Teilnehmer*innen würden das Dashboard 2–3 Mal täglich betrachten, zwei würden es täglich betrachten und eine*r wöchentlich. Die Teilnehmer*innen wurden gefragt, welche Schwierigkeiten sie bei der regelmäßigen Pflege der Konfiguration sehen. Sie konnten darauf mehrere Antworten geben. Für keinen der Teilnehmer*innen war das fehlende Interesse ein Grund. Ein Drittel gab an, dass sie befürchten, die Konfiguration sei zu mühsam für die regelmäßige Durchführung. Zwei Drittel sahen ein Problem in der fehlenden Zeit im laufenden Betrieb und darin, dass es schlichtweg vergessen werden kann. Eine*r der sechs Teilnehmer*innen gab zudem an, dass das Aufsetzen einfach und schnell funktionieren muss und äußerte Bedenken, dass es ab einem bestimmten Punkt unübersichtlich werden kann.

Die Bereitschaft, das System zu pflegen ist überwiegend vorhanden. Für die häufige Betrachtung des Dashboards liegt die Bereitschaft tendenziell sogar noch höher, vorausgesetzt neue Informationen werden gepflegt. Als potenzielle Schwierigkeiten bei der Pflege wurden einige Gründe genannt. Demnach lässt sich die Bereitschaft zur Pflege der Konfiguration durch eine bessere Übersichtlichkeit der Konfigurationsoberfläche und Funktionen, die die Konfiguration erleichtern und beschleunigen, noch erhöhen. Um das Vergessen zu verhindern, sollte eine Erinnerung zur Konfiguration der Auswertung in den Entwicklungsprozess integriert werden.

Konfiguration des Abrufs

Die Fragen 10–13 zur Konfiguration des Abrufs sollen offenlegen, wie gut Abruf-Systeme bereits konfiguriert werden können.

Zwei Drittel der Teilnehmer*innen gaben an, dass sie 10–15 oder sogar über 15 Minuten für die Konfiguration der Abruf-Systeme brauchen. Die übrigen gaben an, 3–6 oder 6–10 Minuten zu brauchen. Für die Hälfte der Teilnehmer*innen war die Konfiguration nicht mühsam. Für einen der sechs Teilnehmer*innen war sie sogar sehr einfach. Zwei Teilnehmer*innen empfanden sie allerdings als mühsam oder sehr mühsam. Die Hälfte der Teilnehmer*innen fand die Konfiguration intuitiv. Die anderen fanden sie noch nicht

oder nur wenig intuitiv. Drei Teilnehmer*innen gaben Vorschläge zur Verbesserung. Sie schlugen vor, mehr Auswahlfelder zu verwenden, um eine bessere Unterstützung bei der Eingabe zu erhalten. Es wurde vorgeschlagen, kurze Beschreibungen direkt an den Feldern zu platzieren, die zum Beispiel auftauchen, wenn der Mauszeiger über das Feld bewegt wird. Auch wurden aussagekräftigere Bezeichner für die Felder gewünscht. Außerdem könne die Hierarchie der Felder durch mehr Einrückung und unterschiedliche Farben klarer gestaltet werden.

Es zeigt sich, dass die Abruf-Konfiguration bereits bedienbar und einsatzbereit ist, aber noch Verbesserungspotenzial in ihr steckt, um sie für alle Anwender*innen intuitiv und angenehm bedienbar zu machen und eine schnellere Konfiguration zu ermöglichen.

Konfiguration der Transformation

Die Fragen 14–17 zur Konfiguration der Transformation geben Aufschluss, wie gut die Konfiguration von Ziel-Entitäten funktioniert.

Ein Drittel der Teilnehmer*innen gab an, 3–6 Minuten für die Konfiguration zu benötigen. Ein weiteres Drittel gab 6–10 Minuten und das übrige Drittel über 15 Minuten an. Zwei Drittel fanden die Konfiguration eher einfach, ein Drittel eher mühsam. Zwei Drittel bewerteten die Konfiguration als intuitiv, die verbleibenden Teilnehmer*innen empfanden sie als wenig oder nicht intuitiv. Drei haben Verbesserungen vorgeschlagen. Zur Verbesserung wurde vorgeschlagen, die verfügbaren Funktionen zur Transformation aus einem Auswahlfeld wählen zu können und einen einfachen Satz zur Verwendung der Funktion anzuzeigen, wenn diese ausgewählt wurde. Zudem wurde empfohlen, aussagekräftigere Namen für die Felder anzuzeigen, sowie Beschreibungen.

Insgesamt betrachtet, ist die Konfiguration der Transformation einsatzfähig, in der Einfachheit und Intuitivität ihrer Bedienung ist jedoch eine Verbesserung gewünscht.

Übersichtsseite

Die Fragen 18–20 werten die Verständlichkeit und Nützlichkeit der Übersichtsseite aus. Fünf der sechs Teilnehmer*innen haben Antworten zu diesen Fragen abgegeben.

Für alle Teilnehmer*innen war die Übersichtsseite verständlich, ein*e Teilnehmer*in (20 %) fand sie etwas weniger verständlich als der Rest. 60 % fanden die Übersicht sehr nützlich, die übrigen haben eine neutrale Wertung abgegeben. Drei Teilnehmer*innen haben Vorschläge zur Verbesserung abgegeben. Eine*r wünscht sich Fehlermeldungen, um Fehler bei der Konfiguration einfacher beheben zu können. Ein*e weitere*r Teilnehmer*in merkt an, dass die Informationen in einem Design dargestellt werden können, es würde aber zum einfachen Verstehen ausreichen. Die Person äußert auch, dass die Übersichtsseite aus ihrer Sicht eingespart werden kann, wenn die Information im Log angezeigt werden. Die dritte Person ist sich unsicher, was verbessert werden kann.

Die Übersichtsseite ist für die Teilnehmer*innen verständlich und für die meisten auch nützlich. Ein ausführlicheres Fehlerhandling ist vorteilhaft. Viele Konfigurationsfehler lassen sich sicher schon durch eine bessere Validierung der Eingabedaten auf der Konfigurationsseite verhindern.

Dashboard

Die Fragen 21–23 sollen aufklären, wie verständlich und informativ das Dashboard empfunden wird. Fünf der sechs Teilnehmer*innen haben Antworten zu diesen Fragen abgegeben.

60 % der Teilnehmer*innen fanden die das Dashboard gut verständlich, die übrigen 40 % fanden es verständlich. 80 % der Teilnehmer*innen fanden es sehr informativ, die übrigen 20 % fanden es informativ. Vorschläge zur Verbesserung des Dashboards wurden nicht abgegeben.

Das Dashboard ist grundsätzlich gut verständlich und informativ, Verbesserungen insbesondere in Bezug auf die Verständlichkeit sind aber zu erwägen. Welche Maßnahmen die Verständlichkeit erhöhen, sollte bei einem längerfristigen Einsatz des Dashboards ergründet werden.

Gesamteindruck

Die Fragen 24–26 sollen abschließend den Gesamteindruck des Systems auf die Teilnehmer*innen ermitteln.

Fünf der sechs Teilnehmer*innen bewerten ihren Gesamteindruck als eher gut, eine Person als gut. Die Hälfte der Teilnehmer*innen würden das System wahrscheinlich weiterempfehlen, ein Drittel sehr wahrscheinlich und eine*r der sechs Teilnehmer*innen ist dem gegenüber neutral eingestellt. Zwei Teilnehmer*innen haben weitere Kommentare abgegeben. Ein*e weitere*r gab an, sie oder er fände eine Gegenüberstellung des Aufwands für ein neues Feature im Vergleich zum generierten Umsatz durch ein Feature interessant. Die Informationen zu den Aufwänden für ein Feature können von der Projektmanagement-Software Jira erhalten werden. Eine weitere Person gab an, sie benötige mehr Zeit mit dem System, um weitere Verbesserungsvorschläge zu geben. Das Dashboard sehe gut aus und das System könne auf viele Arten hilfreich sein. Die Person hat auch darauf hingewiesen, dass die Anmeldedaten auf dem System zusätzlich gesichert sein sollten, zum Beispiel durch Verschlüsselung.

Die Teilnehmer*innen gaben nach der Evaluationsphase eine überwiegend positive Bewertung ab. Zahlreiche Rückmeldungen an Verbesserungsvorschlägen zeigen aber auch, dass das System noch viel Potenzial zur Verbesserung hat.

7 Fazit

Dieses Kapitel fasst die Ergebnisse der Arbeit zusammen. Im Anschluss wird ein Ausblick gegeben, welche Möglichkeiten es zur Erweiterung gibt.

7.1 Zusammenfassung

Im Rahmen dieser Arbeit wurde ein System konzipiert und entwickelt, welches die Mehrwerte von Features in Onlineshops auswertet und visuell darstellt. Informationen zu den Features werden dazu von verschiedenen Systemen abgerufen und verarbeitet. Die Ergebnisse werden auf einem Dashboard dargestellt, um Mitarbeiter*innen zu motivieren und Erkenntnisse zu gewinnen.

Es wurde insbesondere Wert darauf gelegt, dass der Abruf benötigter Informationen und die Auswertung der Informationen auf eine einfache und schnelle Weise konfiguriert werden können. So soll es möglich sein, bereits im Entwicklungsprozess die Auswertungen für implementierte Features zu konfigurieren und ab dem Release des Features verfolgen zu können. Hierfür wurde eine Struktur für die Konfiguration konzipiert und eine Weboberfläche zur Durchführung der Konfiguration entwickelt.

Der Datenabruf ist so umgesetzt, dass die Systeme, von denen Daten bezogen werden, nicht zu stark belastet werden. Wenn größere Datenmengen abgerufen werden sollen, wird die Abfrage aufgeteilt und zeitlich gestreckt. Eine Kopie der abgerufenen Daten wird nach dem Konzept des Data Lake gespeichert, sodass keine Daten doppelt abgerufen werden müssen, selbst wenn neue Auswertungen erstellt werden.

Zur Erprobung des Systems wurden verschiedene Auswertungen angelegt. Dazu wurden zunächst anhand des GQM-Modells die abzufragenden Metriken ermittelt. Die Bestelldaten aus einem Onlineshop wurden zusammen mit den Daten aus einem Projektmanagement-System verarbeitet, um den Umsatzverlauf in Zusammenhang mit Releases darzustellen.

In einer weiteren Auswertung wurden Daten, die eine Bestellung mit einem Feature in Zusammenhang bringen lassen, vom Onlineshop abgefragt und die Umsätze nach Bestellungen mit diesem Feature gefiltert. So konnte ein Umsatzverlauf für dieses Feature dargestellt werden, dessen Werte als Faktor für den Mehrwert des Features betrachtet werden können.

Zur Evaluation wurde das System auf einem Server für die Teilnehmer*innen einer Umfrage bereitgestellt. Diese haben Auswertungen konfiguriert und die Ergebnisse auf dem Dashboard betrachtet, um eine Bewertung vorzunehmen. Darüber hinaus haben sie wertvolle Anmerkungen zur Verbesserung des Systems angegeben.

Insgesamt wurde ein einsatzfähiges System entwickelt, um anhand von Konfigurationen visuelle Auswertungen für Features zu erstellen. Die Tester*innen haben das System grundsätzlich positiv bewertet, einige hatten aber Verbesserungsvorschläge, um die Konfiguration intuitiver zu gestalten. Um es im laufenden Entwicklungsbetrieb reibungslos einzusetzen, können Teile des Systems weiter verbessert werden.

7.2 Ausblick

Um das System im Entwicklungsprozess zu integrieren, sollte die in den Grundlagen beschriebene Technik zur Auswahl der Daten (vgl. *2.6.1 Auswahl der Daten*) etabliert werden, sodass die Entwickler*innen selbständig die Konfiguration für alle relevanten Features durchführen können.

Es gibt verschiedene Möglichkeiten das System zu erweitern und zu verbessern.

Daten aus Google Analytics auswerten. Aus Google Analytics können Informationen über Bestellungen und Umsätze abgerufen werden (vgl. *2.8 Google Analytics*). Die Daten stammen von Skripten, welche beim Aufruf der Bestellbestätigungsseite nach dem Abschluss der Bestellung ausgeführt werden. Diese senden dann die Information an Google Analytics. Retouren werden von Google Analytics nicht festgestellt. Durch Zusammenführung der Umsatzinformationen aus Google Analytics und Magento kann ausgewertet werden, wie groß die Abweichungen, beispielsweise durch nicht getrackte Retouren, sind. Diese Abweichungen können in einem Diagramm als Umsatzdifferenz pro Zeit dargestellt werden. Darüber hinaus kann durch Daten zur Nutzung von Features und Kaufabschlüssen die Konversionsrate in Zusammenhang mit Features bestimmt werden.

So kann die Frage beantwortet werden, wie viele Feature-Nutzer*innen auch gekauft haben. Weitere Auswertung bezüglich Nutzungshäufigkeit von Features und Kaufabbrüche, sowie verbrachte Zeit mit einem Feature können anhand der Trackingdaten ausgewertet werden.

Benachrichtigungen bei starken Veränderungen. Um bei auffälligen Schwankungen von Werten die Aufmerksamkeit auf das Dashboard zu ziehen, können Benachrichtigungen versendet werden. Diese können sowohl für negative Entwicklungen, um beispielsweise Fehler aufzudecken, oder als Erfolgsmeldungen bei positiven Entwicklungen gesendet werden. Regeln für das Versenden von Benachrichtigungen können in Grafana festgelegt werden (vgl. *2.5.4 Alerts*).

Auswertung der Besuche. Die Besuche eines Onlineshops können nach Plattform, also dem genutzten Gerät, Betriebssystem und Browser, ausgewertet werden. Zum einen ist die Messung der Anzahl an Besuchen pro Plattform interessant, aber auch die Umsätze können für Besuche mit Kaufabschluss kumuliert werden. So ist es möglich, Optimierungen für bestimmte Plattformen vorzunehmen. Die gleiche Auswertung kann für die Besuche gemacht werden, abhängig davon, ob die Besucher*innen mit ihrem Kundenkonto eingeloggt sind oder nicht.

Umsatzvergleich zwischen Releases. Um eine Tendenz darstellen zu können, kann der Umsatz zwischen den Releases verglichen werden. Es können so die Einnahmen pro Zeiteinheit vor und nach dem letzten Release ausgewertet werden und angezeigt werden, ob der Wert im Vergleich zu vorherigen Releases steigt oder sinkt.

Vergleich von Aufwand und Mehrwert. Um eine genauere Einschätzung zu erhalten, wie lohnenswert die Umsetzung eines Features ist, können Umsetzungsdauer und festgestellter Mehrwert eines Features gegenübergestellt werden. Die Umsetzungsdauer kann von Projektmanagement-Systemen erhalten werden, die für die Zeiterfassung genutzt werden.

Automatische Aktualisierung. Die Aktionen zum Abruf und zur Transformation können zeitgesteuert ausgeführt werden, um Aufwand einzusparen (vgl. *3.3.1 Automatische Aktualisierung*).

Konfiguration vereinfachen. Um die Konfiguration benutzerfreundlicher zu gestalten, können aussagekräftigere Namen für die Felder angezeigt werden. Außerdem können Beschreibungen direkt an den Feldern angezeigt werden, wenn der Mauszeiger darüber

bewegt wird. Durch spezifische Auswahlfelder, wie beispielsweise die Auswahl eines Datums aus einer Kalenderansicht oder Vorschläge für verfügbare Funktionen, kann die Konfiguration vereinfacht und Fehler verhindert werden. Eine Validierung der Eingaben mit aussagekräftiger Fehlerbeschreibung verhindert ebenso fehlerhafte Eingaben.

Konfigurationsassistent. Ein weiterer Schritt zur Vereinfachung der Konfiguration ist ein Assistent, der durch die Schritte des GQM-Modells führt, um die abzurufenden Daten zu bestimmen. Benutzer*innen müssen nur noch Fragen beantworten und die Konfiguration wird generiert.

Voraussagen treffen. Anhand von vorhandenen Daten können Zukunftsprognosen berechnet werden. Durch selbstlernende Prozesse können die Prognosen verbessert werden, indem die Voraussagen mit tatsächlich eingetroffenen Werten verglichen werden. Durch die Voraussagen können auftretende Verläufe gegebenenfalls noch früher erkannt werden.

Erweiterte Absicherung von Anmeldedaten. Auf das System haben nur autorisierte Personen Zugriff und Daten werden verschlüsselt übertragen. Da viele Anmeldedaten an einem Ort gespeichert werden, sollte die Sicherheit erhöht werden, indem die Daten verschlüsselt auf dem System gespeichert werden. So sind die Daten sicher, selbst wenn sich unbefugte Personen Zugang zum System verschaffen. Eine weitere Möglichkeit ist, einen Passwortmanager zu verknüpfen, der bereits für die Verwaltung der Passwörter benutzt wird. So müssen die Anmeldedaten nicht an einem weiteren Ort gespeichert werden und können stattdessen bei Bedarf abgerufen werden.

Vertiefende Evaluation. Um das System ausführlicher zu evaluieren, sollte es über einen längeren Zeitraum und von einer größeren Personengruppe getestet werden.

Literaturverzeichnis

[Aden 2012] Aden, Timo: *Google Analytics Implementieren. Interpretieren. Profitieren.* 3., aktualisierte und erw. Aufl. Hanser Verlag, 2012. – URL http://www.hanser-elibrary.com/doi/book/10.3139/9783446432826

[Denney u. a. 2016] Denney, Michael J. ; Long, Dustin M. ; Armistead, Matthew G. ; Anderson, Jamie L. ; Conway, Baqiyyah N.: Validating the extract, transform, load process used to populate a large clinical research database. In: *International journal of medical informatics* 94 (2016), S. 271–274. – URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5556907/

[Divya und Goyal 2015] Divya, Manda S. ; Goyal, Shiv K.: ElasticSearch 'An advanced and quick search technique to handle voluminous data'. In: *COMPUSOFT: An International Journal of Advanced Computer Technology* 2 (2015), Nov., Nr. 6. – URL https://ijact.joae.org/index.php/ijact/article/view/380

[Dixon 2010] Dixon, James: *Pentaho, Hadoop, and Data Lakes.* 2010. – URL https://jamesdixon.wordpress.com/2010/10/14/pentaho-hadoop-and-data-lakes/. – Zugriffsdatum: 11.02.2021

[Few 2006] Few, Stephen: *Information Dashboard Design: The Effective Visual Communication of Data.* O'Reilly Media, Inc., 2006. – URL https://dl.acm.org/doi/10.5555/1206491

[Gartner, Inc. 2020] Gartner, Inc.: *Gartner Magic Quadrant for Digital Commerce.* 2020. – URL https://www.gartner.com/en/documents/3989439. – Zugriffsdatum: 02.07.2021

[Grafana Labs 2021a] Grafana Labs: *Alerts.* 2021. – URL https://grafana.com/docs/grafana/latest/alerting/. – Zugriffsdatum: 29.06.2021

[Grafana Labs 2021b] Grafana Labs: *Annotations*. 2021. – URL https://grafana.com/docs/grafana/latest/dashboards/annotations/. – Zugriffsdatum: 29.06.2021

[Grafana Labs 2021c] Grafana Labs: *Dashboards*. 2021. – URL https://grafana.com/docs/grafana/latest/dashboards/. – Zugriffsdatum: 29.06.2021

[Grafana Labs 2021d] Grafana Labs: *Data sources*. 2021. – URL https://grafana.com/docs/grafana/latest/datasources/. – Zugriffsdatum: 29.06.2021

[Grafana Labs 2021e] Grafana Labs: *Grafana basics*. 2021. – URL https://grafana.com/docs/grafana/latest/basics/. – Zugriffsdatum: 29.06.2021

[Grafana Labs 2021f] Grafana Labs: *HTTP API*. 2021. – URL https://grafana.com/docs/grafana/latest/http_api/. – Zugriffsdatum: 29.06.2021

[Grafana Labs 2021g] Grafana Labs: *Panels*. 2021. – URL https://grafana.com/docs/grafana/latest/panels/. – Zugriffsdatum: 29.06.2021

[Grafana Labs 2021h] Grafana Labs: *Queries*. 2021. – URL https://grafana.com/docs/grafana/latest/panels/queries/. – Zugriffsdatum: 29.06.2021

[Janes u. a. 2013] Janes, Andrea ; Sillitti, Alberto ; Succi, Giancarlo: Effective dashboard design. In: *Cutter IT Journal* 26 (2013), 01, S. 17–24. – URL https://www.researchgate.net/publication/286996830

[Kollewe 2016] Kollewe, Tobias ; Keukert, Michael (Hrsg.): *Praxiswissen E-Commerce Das Handbuch für den erfolgreichen Onlineshop*. 2. Aufl. O'Reilly Verlag, 2016 (Basics). – URL http://gbv.eblib.com/patron/FullRecord.aspx?p=4652640

[Kuć 2014] Kuć, Rafał ; Rogoziński, Marek (Hrsg.): *Elasticsearch Server A practical guide to building fast, scalable, and flexible search solutions with clear and easy-to-understand examples*. Second Edition. Packt Publishing, 2014 (Community Experience Distilled). – URL https://www.packtpub.com/product/elasticsearch-server-second-edition/9781783980529

[Launix 2016] Launix: *Daten geschickt sammeln - Big Data für Jedermann*. 2016. – URL https://launix.de/launix/daten-geschickt-sammeln-big-data-fuer-jedermann/. – Zugriffsdatum: 11.02.2021

[Louwers 2012] Louwers, Johan: *sub transactional big-data and data analysis*. 2012. – URL http://johanlouwers.blogspot.com/2012/02/sub-transactional-big-data-and-data.html. – Zugriffsdatum: 11.02.2021

[Oracle Corp. 2002] Oracle Corp.: *Data Warehousing Concepts*. 2002. – URL https://docs.oracle.com/cd/B10501_01/server.920/a96520/concept.htm. – Zugriffsdatum: 26.06.2021

[Papp 2019] Papp, Stefan ; Weidinger, Wolfgang (Hrsg.): *Handbuch Data Science mit Datenanalyse und Machine Learning Wert aus Daten generieren*. Hanser, 2019 (Hanser eLibrary). – URL https://www.hanser-elibrary.com/doi/book/10.3139/9783446459755

[Starke und Hruschka 2005] Starke, Gernot ; Hruschka, Peter: *arc42 - the template for software architecture documentation and communication*. 2005. – URL https://github.com/arc42/arc42-template. – Zugriffsdatum: 02.06.2021

[Strengholt 2020] Strengholt, Piethein ; Safari, an O'Reilly Media C. (Hrsg.): *Data Management at Scale*. 1st edition. O'Reilly Media, Inc., 2020. – URL https://learning.oreilly.com/library/view/-/9781492054771/?ar

[Tank u. a. 2010] Tank, Darshan M. ; Ganatra, Amit ; Kosta, Y.P. ; Bhensdadia, C.K.: Speeding ETL Processing in Data Warehouses Using High-Performance Joins for Changed Data Capture (CDC). In: *2010 International Conference on Advances in Recent Technologies in Communication and Computing*, URL https://ieeexplore.ieee.org/abstract/document/5656810. – Zugriffsdatum: 20.06.2021, 2010, S. 365–368

[Tilkov und Vinoski 2010] Tilkov, Stefan ; Vinoski, Steve: Node.js: Using JavaScript to Build High-Performance Network Programs. In: *IEEE Internet Computing* 14 (2010), Nr. 6, S. 80–83. – URL https://ieeexplore.ieee.org/abstract/document/5617064

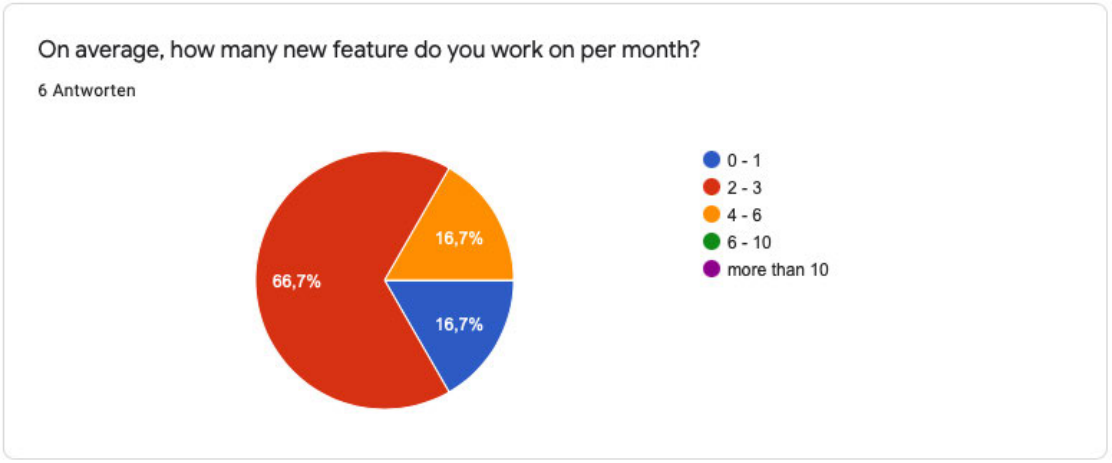
[Wirdemann 2017] Wirdemann, Ralf ; Mainusch, Johannes (Hrsg.): *Scrum mit User Stories*. 3., erweiterte Auflage. Hanser, 2017. – URL https://www.hanser-elibrary.com/doi/book/10.3139/9783446450776

A Anhang

Fragebogen

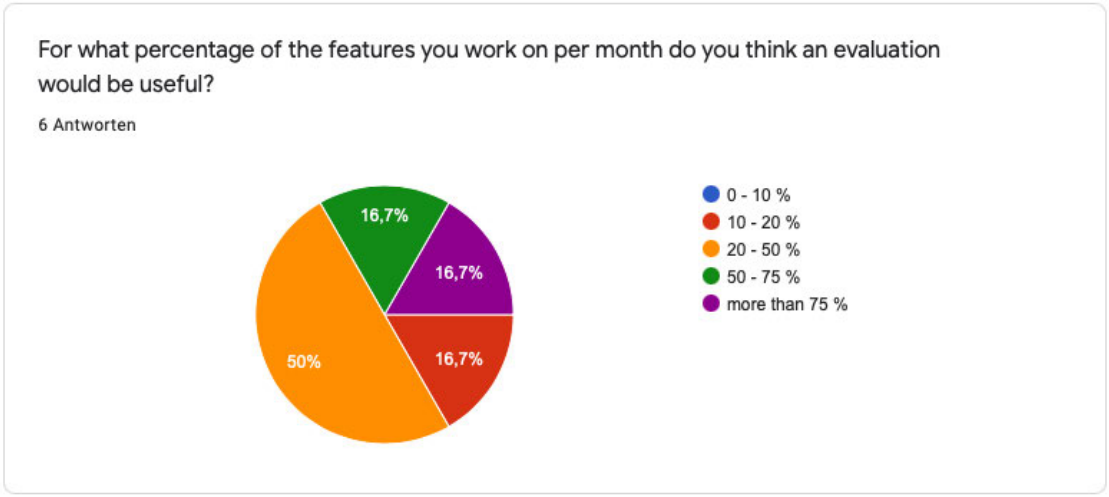
Die Auswertung des Fragebogens wird in *Abschnitt 6.2* erläutert. Der Fragebogen wurde mit der Umfrageverwaltungssoftware *Google Formulare* erstellt.

Motivation



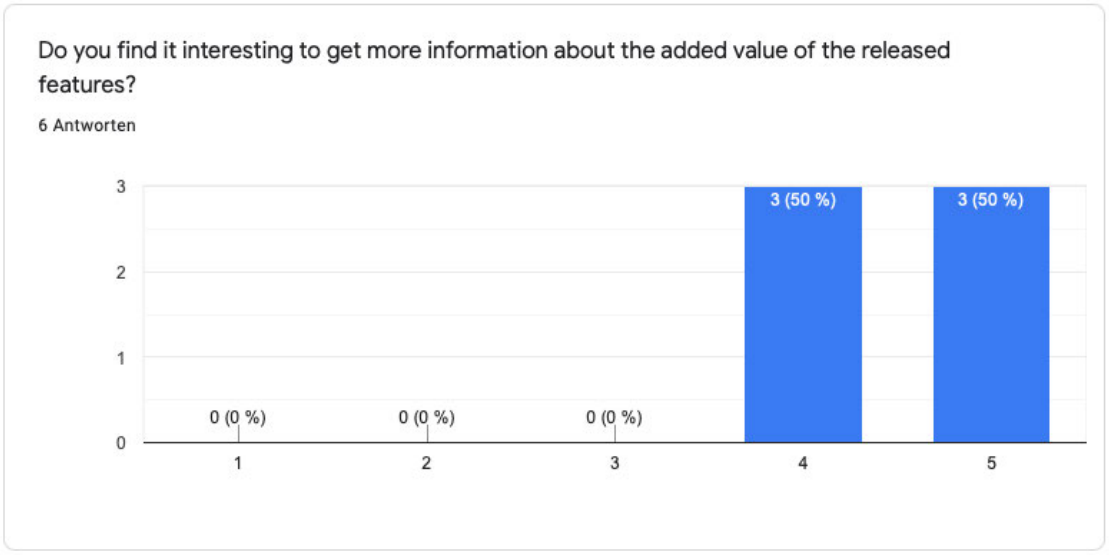
An wie vielen neuen Features arbeiten Sie im Durchschnitt pro Monat?

Abbildung A.1: Frage 1 (Motivation)



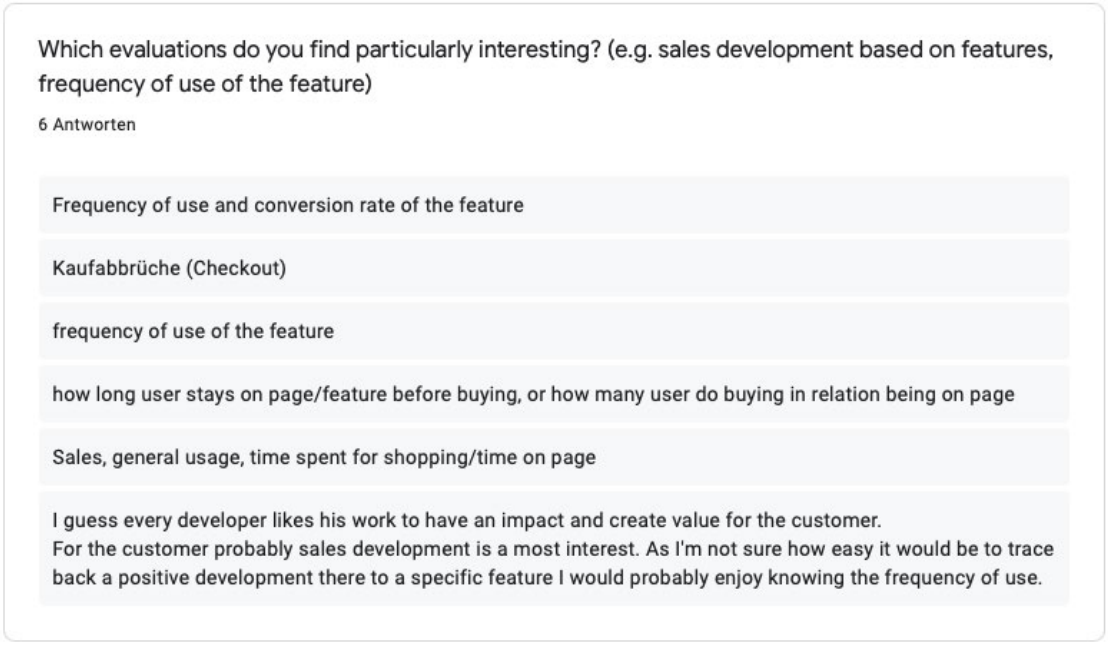
Für wie viel Prozent der Features, an denen Sie pro Monat arbeiten, wäre Ihrer Meinung nach eine Bewertung sinnvoll?

Abbildung A.2: Frage 2 (Motivation)



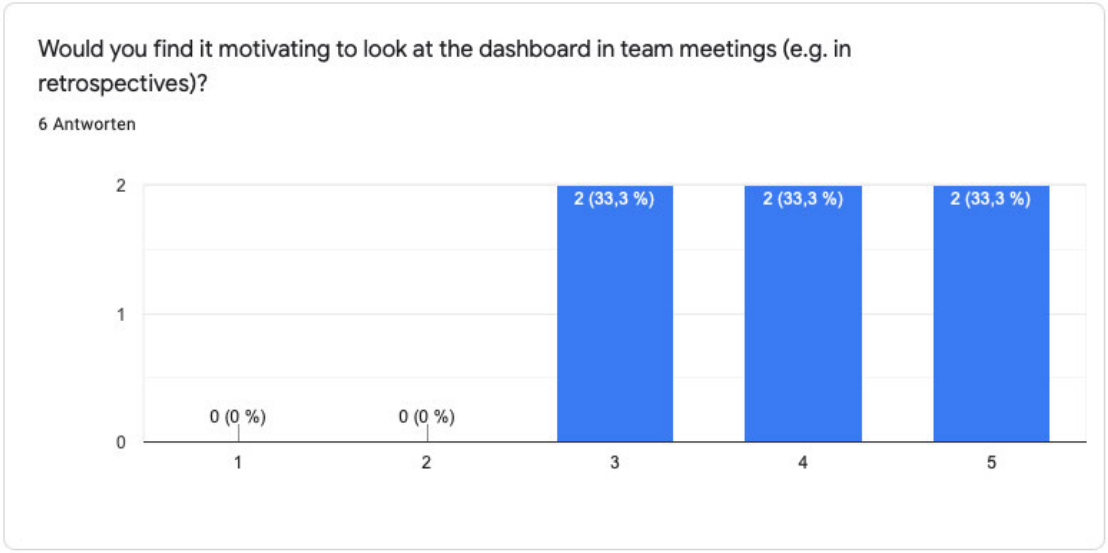
Finden Sie es interessant, mehr Informationen über den Mehrwert der bereitgestellten Features zu erhalten? (1 = Nein, überhaupt nicht – 5 = Ja, sehr interessant)

Abbildung A.3: Frage 3 (Motivation)



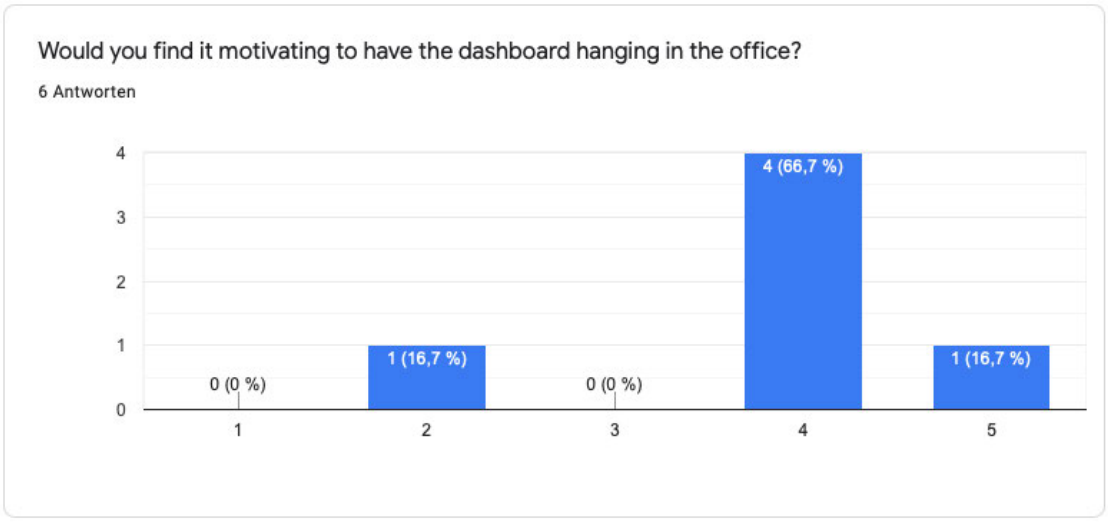
Welche Auswertungen finden Sie besonders interessant? (z. B. Umsatzentwicklung in Abhängigkeit von Features, Häufigkeit der Nutzung eines Features)

Abbildung A.4: Frage 4 (Motivation)



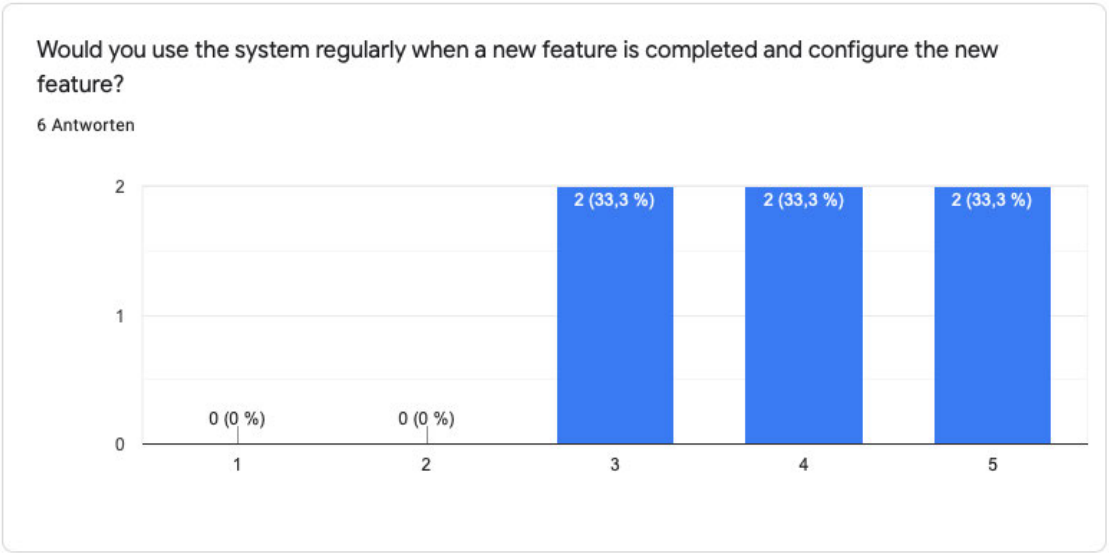
Würden Sie es als motivierend empfinden, sich das Dashboard in Teamsitzungen (z. B. in Retrospektiven) anzusehen? (1 = Nein, überhaupt nicht – 5 = Ja, sehr motivierend)

Abbildung A.5: Frage 5 (Motivation)



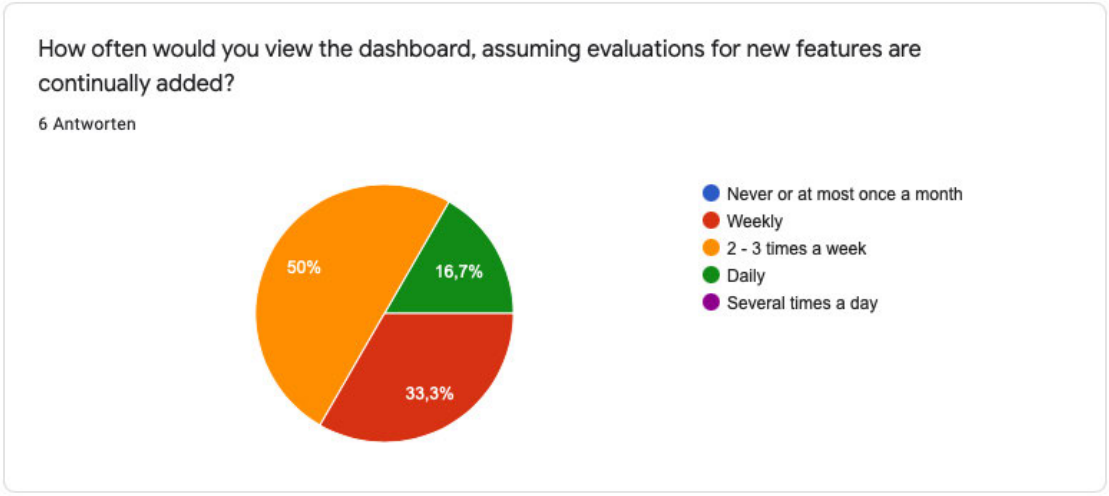
Würden Sie es als motivierend empfinden, das Dashboard im Büro hängen zu haben? (1 = Nein, überhaupt nicht – 5 = Ja, sehr motivierend)

Abbildung A.6: Frage 6 (Motivation)



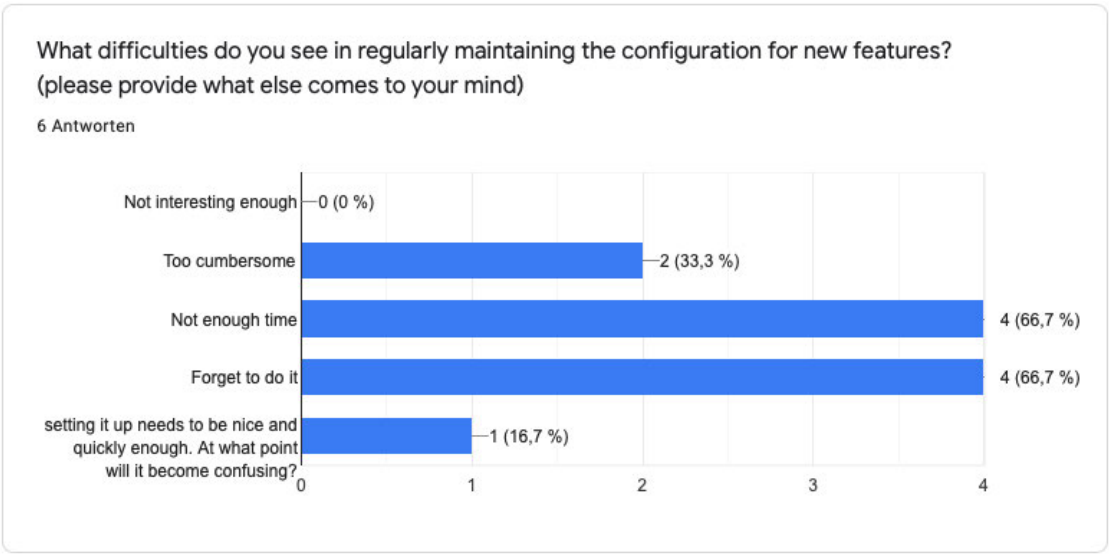
Würden Sie das System regelmäßig nutzen, wenn eine neue Funktion fertiggestellt ist, und die neue Funktion konfigurieren? (1 = Nein, niemals – 5 = Ja, definitiv)

Abbildung A.7: Frage 7 (Motivation)



Wie oft würden Sie sich das Dashboard ansehen, wenn man davon ausgeht, dass regelmäßig Auswertungen für neue Features hinzugefügt werden?

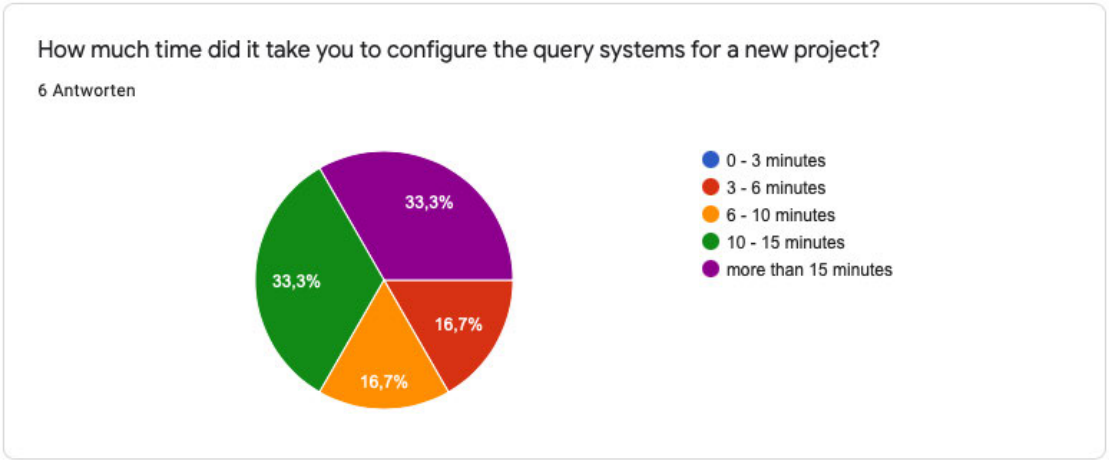
Abbildung A.8: Frage 8 (Motivation)



Welche Schwierigkeiten sehen Sie bei der regelmäßigen Pflege der Konfiguration für neue Features? (bitte geben Sie an, was Ihnen sonst noch einfällt)

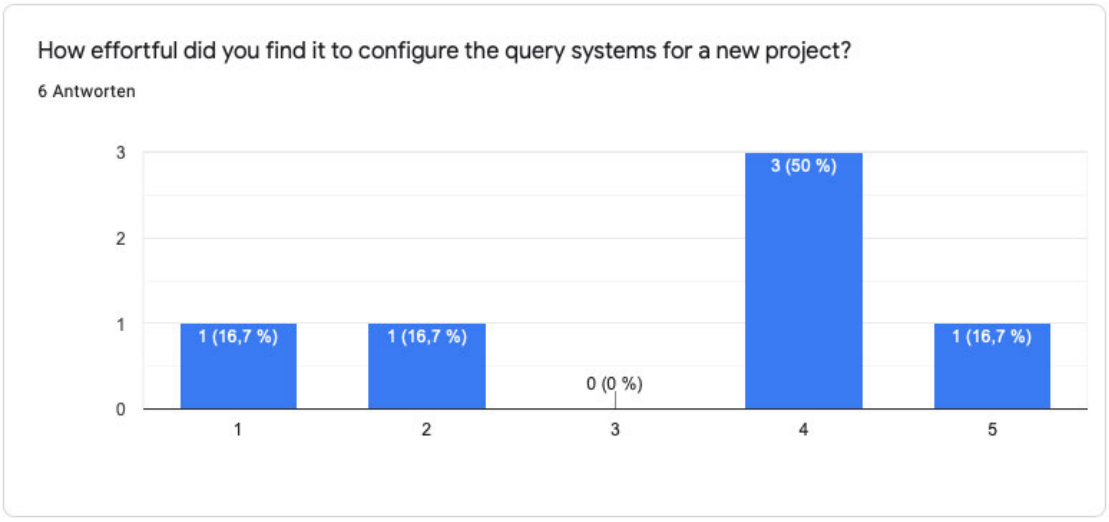
Abbildung A.9: Frage 9 (Motivation)

Konfiguration des Abrufs



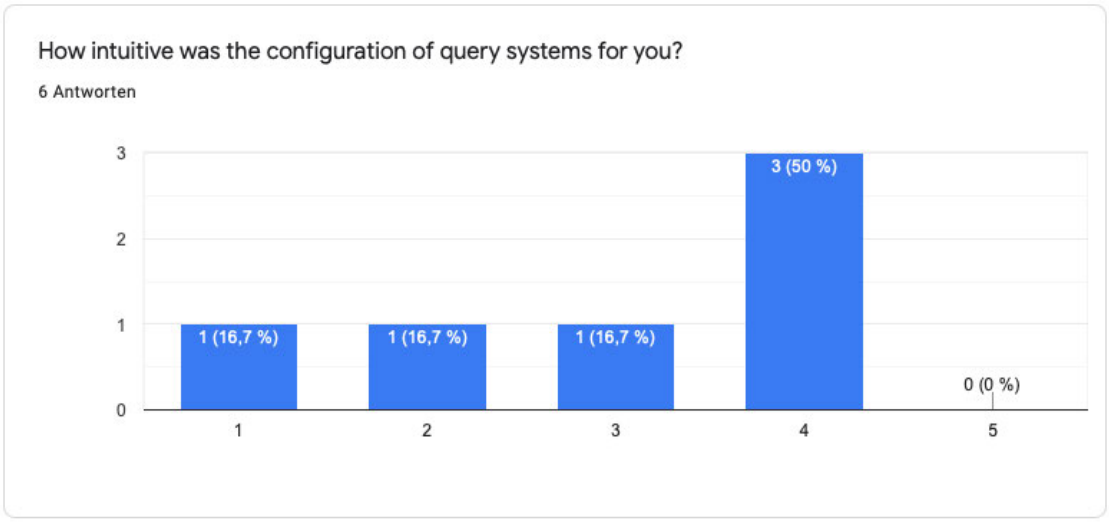
Wie viel Zeit haben Sie für die Konfiguration der Abruf-Systeme für ein neues Projekt benötigt?

Abbildung A.10: Frage 10 (Konfiguration des Abrufs)



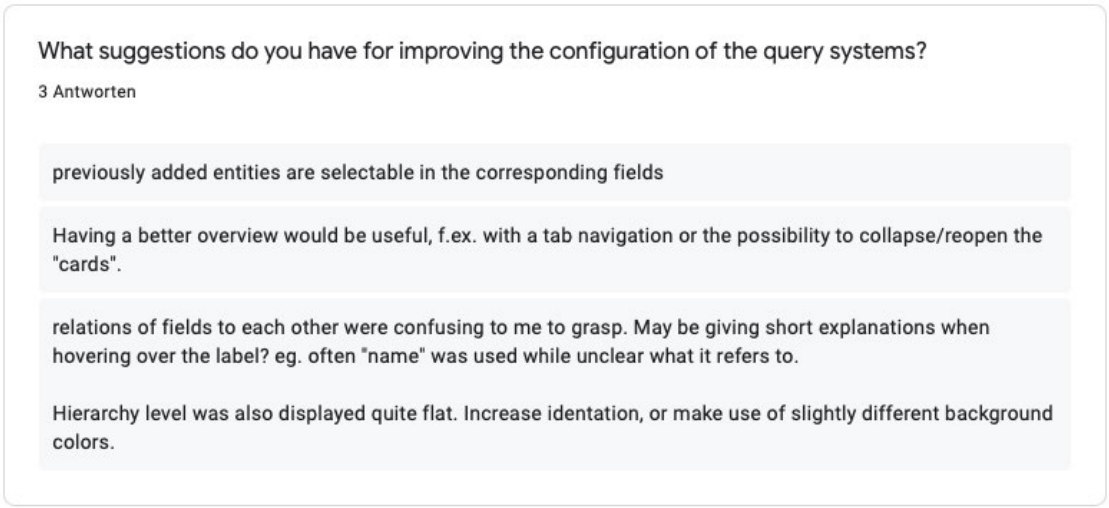
Wie mühsam war es für Sie, die Abruf-Systeme für ein neues Projekt zu konfigurieren? (1 = Sehr mühsam – 5 = Sehr einfach)

Abbildung A.11: Frage 11 (Konfiguration des Abrufs)



Wie intuitiv war die Konfiguration der Abruf-Systeme für Sie?
(1 = Nicht intuitiv – 5 = Sehr intuitiv)

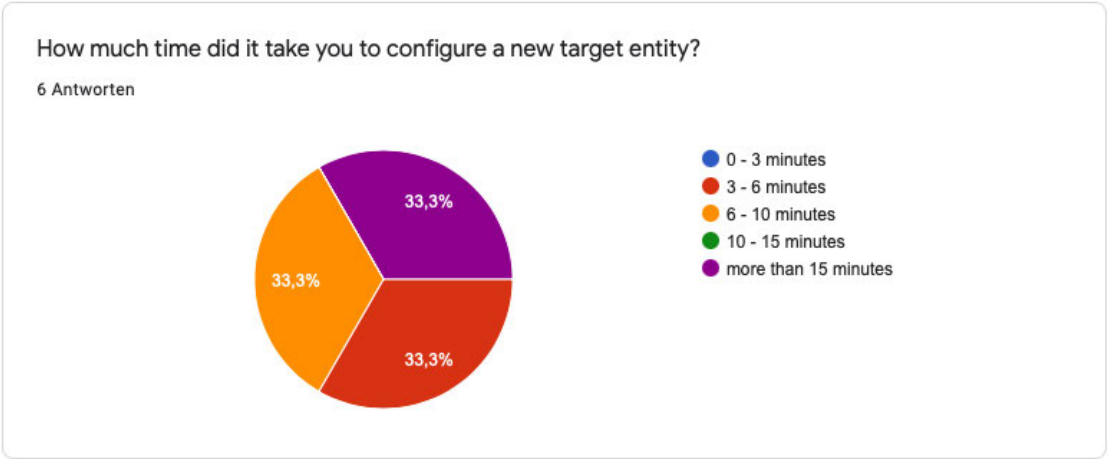
Abbildung A.12: Frage 12 (Konfiguration des Abrufs)



Welche Vorschläge haben Sie zur Verbesserung der Konfiguration der Abruf-Systeme?

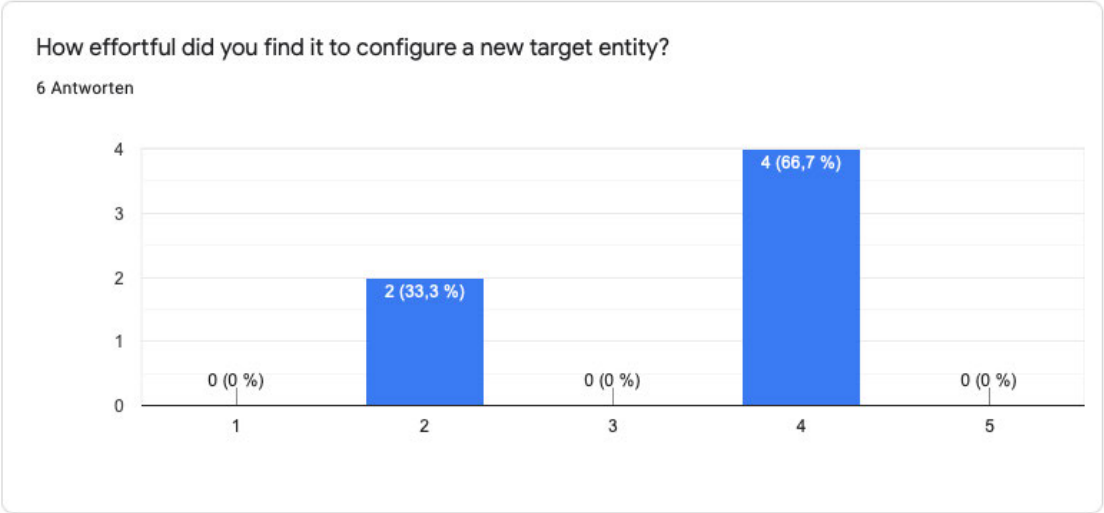
Abbildung A.13: Frage 13 (Konfiguration des Abrufs)

Konfiguration der Transformation



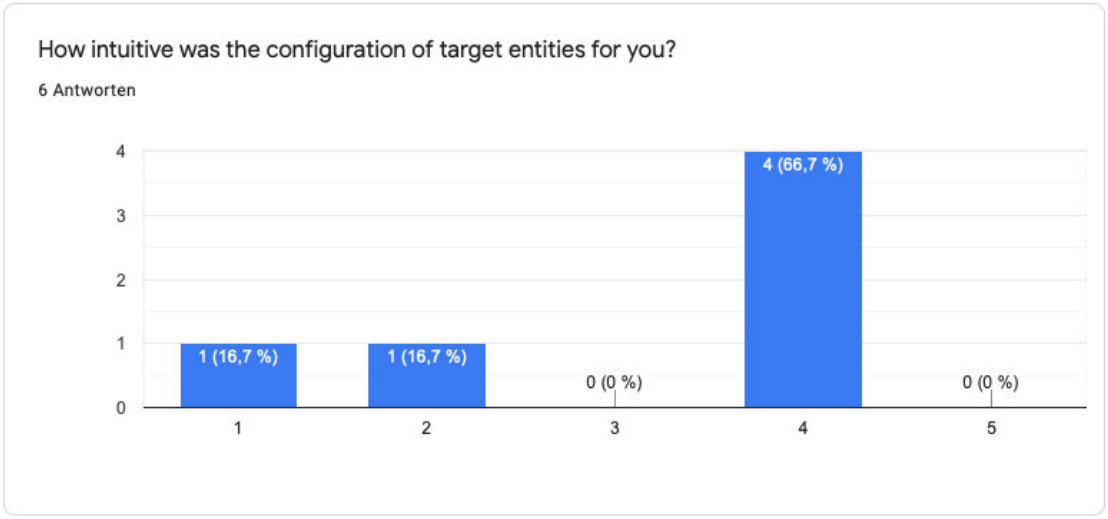
Wie viel Zeit haben Sie für die Konfiguration einer neuen Ziel-Entität benötigt?

Abbildung A.14: Frage 14 (Konfiguration der Transformation)



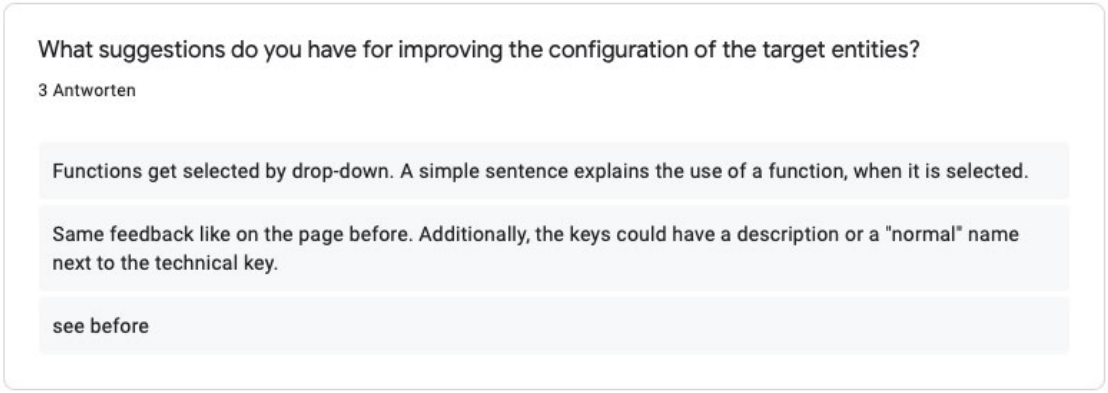
Wie mühsam war es für Sie, eine neue Ziel-Entität zu konfigurieren?
(1 = Sehr mühsam – 5 = Sehr einfach)

Abbildung A.15: Frage 15 (Konfiguration der Transformation)



Wie intuitiv war die Konfiguration der Ziel-Entitäten für Sie?
(1 = Nicht intuitiv – 5 = Sehr intuitiv)

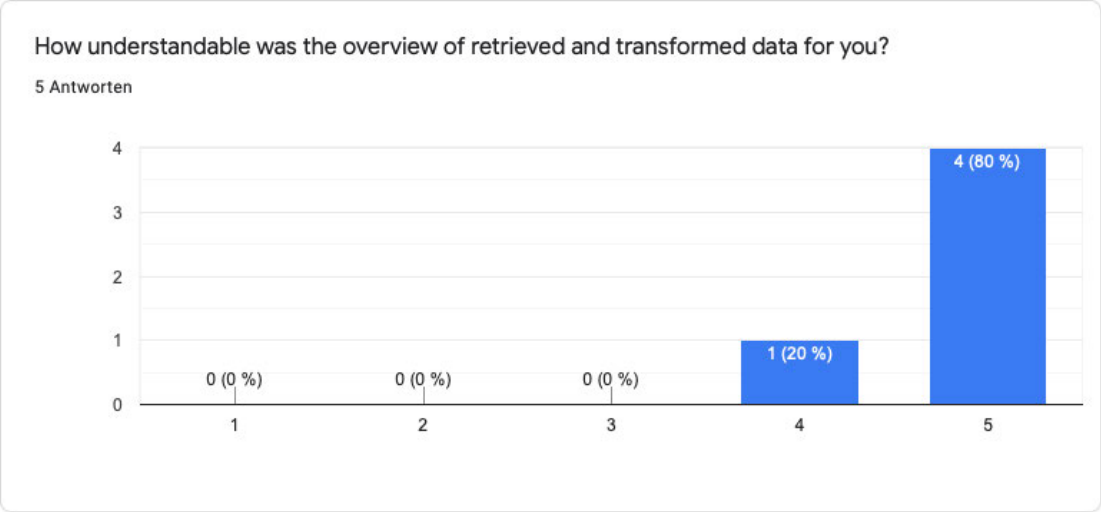
Abbildung A.16: Frage 16 (Konfiguration der Transformation)



Welche Vorschläge haben Sie zur Verbesserung der Konfiguration der Ziel-Entitäten?

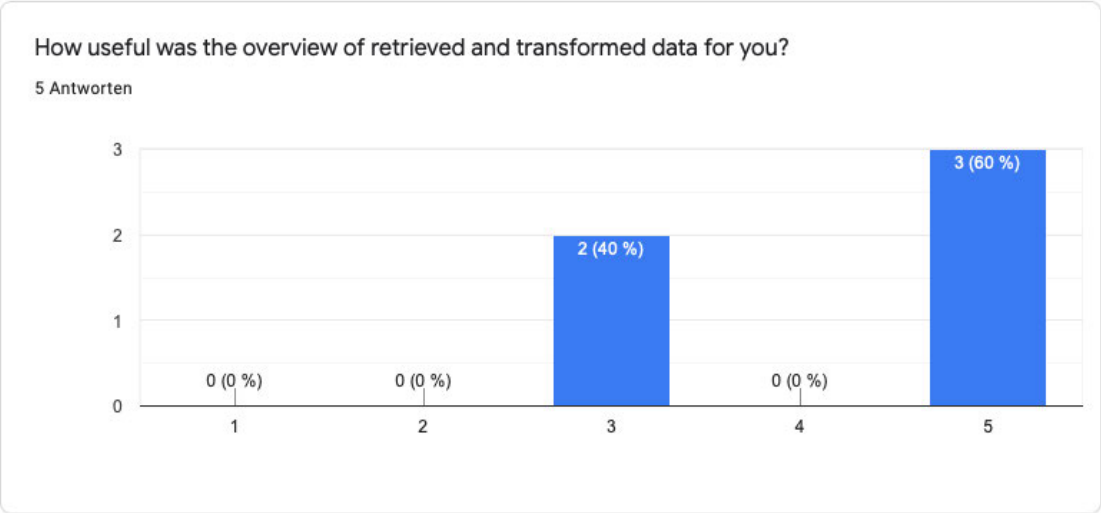
Abbildung A.17: Frage 17 (Konfiguration der Transformation)

Übersichtsseite



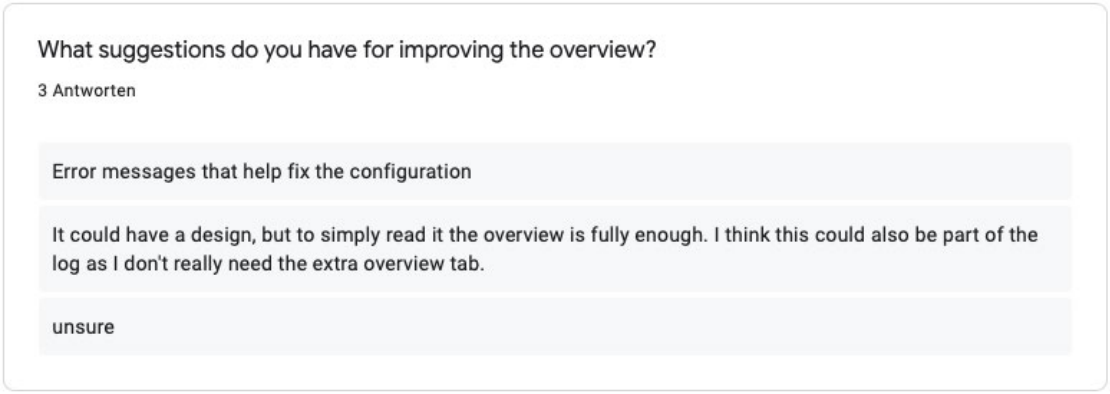
Wie verständlich war für Sie die Übersicht über die abgerufenen und transformierten Daten? (1 = Schwer zu verstehen – 5 = Gut zu verstehen)

Abbildung A.18: Frage 18 (Übersichtsseite)



Wie nützlich war die Übersicht über die abgerufenen und transformierten Daten für Sie? (1 = Nicht nützlich – 5 = Sehr nützlich)

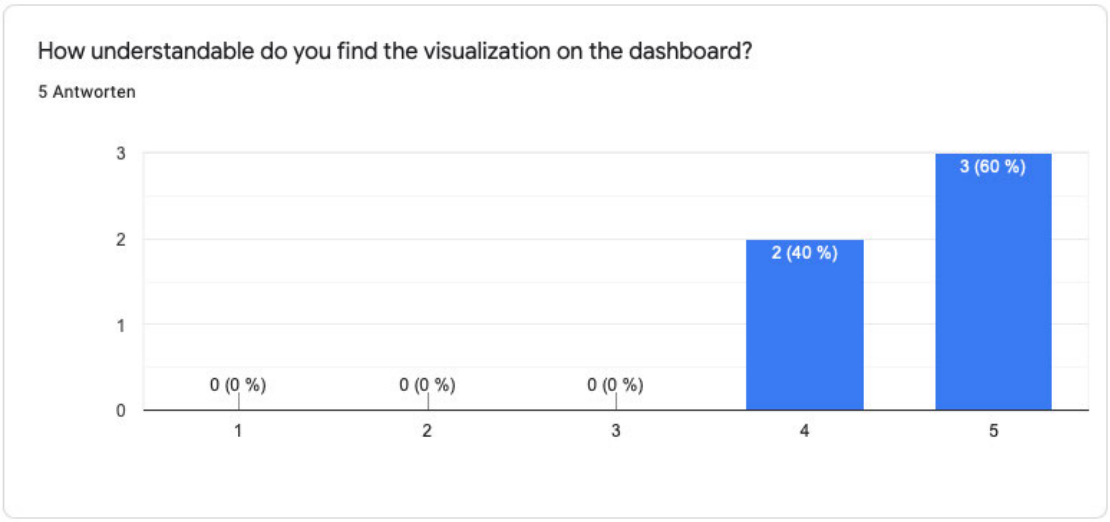
Abbildung A.19: Frage 19 (Übersichtsseite)



Welche Vorschläge haben Sie zur Verbesserung der Übersichtsseite?

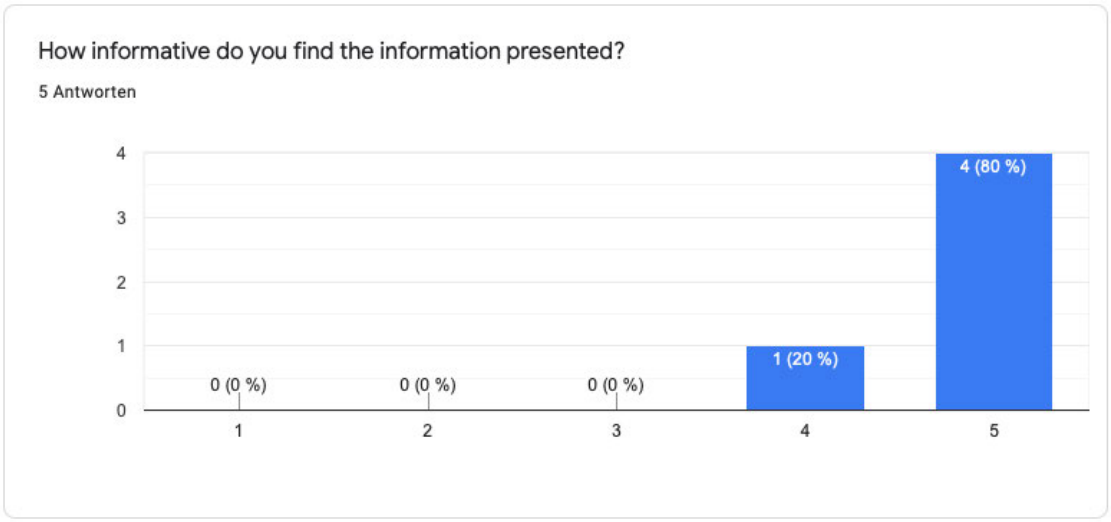
Abbildung A.20: Frage 20 (Übersichtsseite)

Dashboard



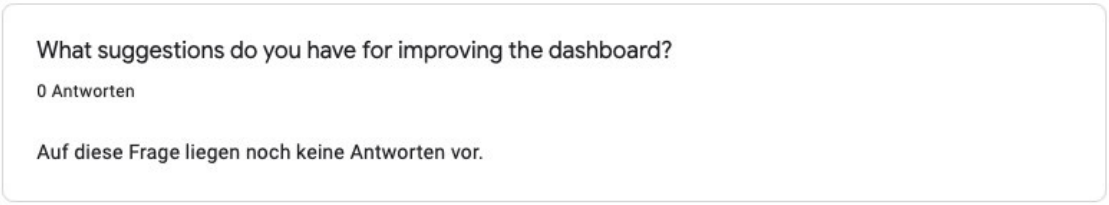
Wie verständlich finden Sie die Visualisierung auf dem Dashboard?
(1 = Schwer zu verstehen – 5 = Gut zu verstehen)

Abbildung A.21: Frage 21 (Dashboard)



Wie informativ finden Sie die präsentierten Informationen?
(1 = Nicht informativ – 5 = Sehr informativ)

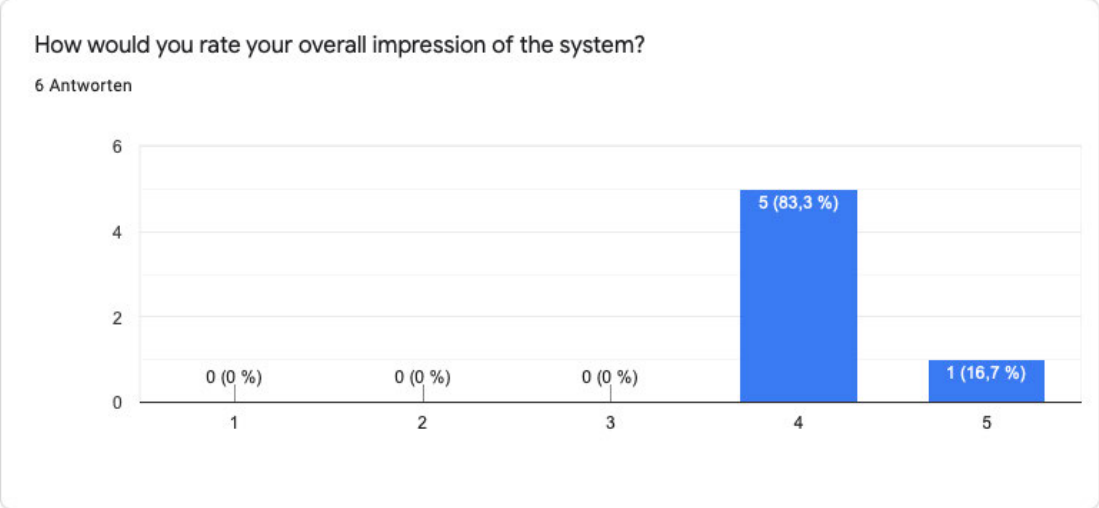
Abbildung A.22: Frage 22 (Dashboard)



Welche Vorschläge haben Sie zur Verbesserung des Dashboards?

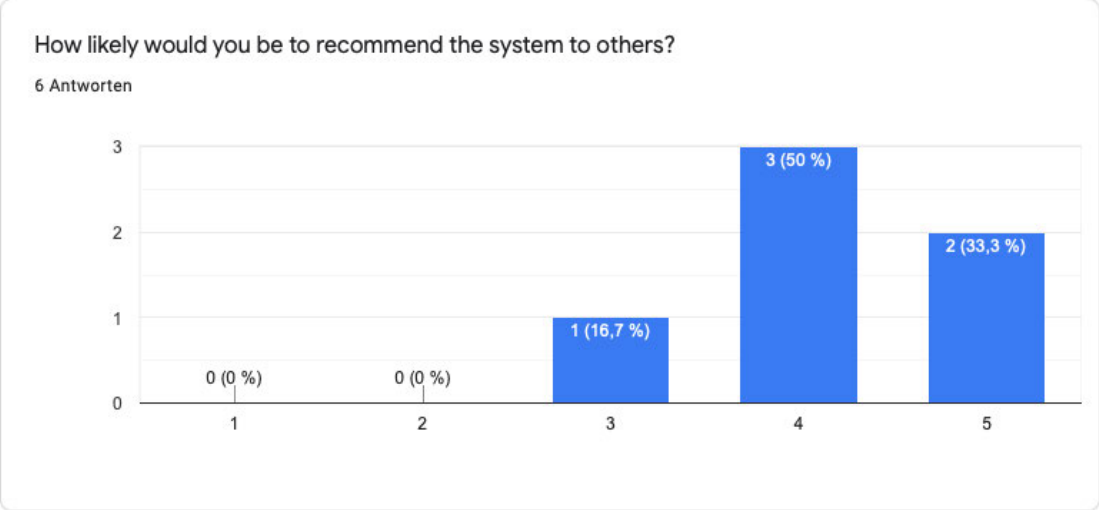
Abbildung A.23: Frage 23 (Dashboard)

Gesamteindruck



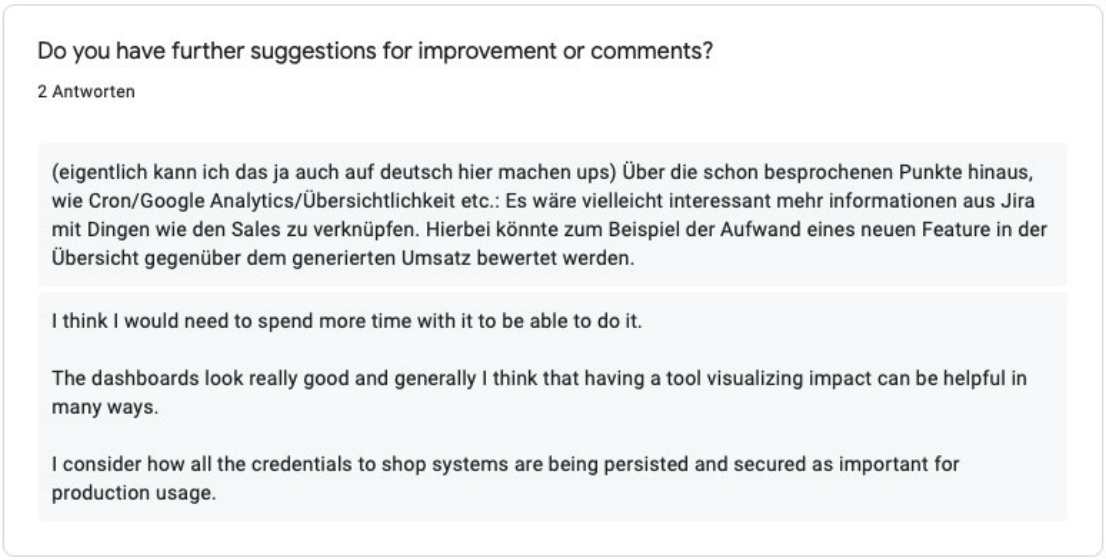
Wie würden Sie Ihren Gesamteindruck des Systems beschreiben?
(1 = Schlecht – 5 = Gut)

Abbildung A.24: Frage 24 (Gesamteindruck)



Wie wahrscheinlich ist es, dass Sie das System weiterempfehlen?
(1 = Unwahrscheinlich – 5 = Sehr wahrscheinlich)

Abbildung A.25: Frage 25 (Gesamteindruck)



Haben Sie weitere Verbesserungsvorschläge oder Kommentare?

Abbildung A.26: Frage 26 (Gesamteindruck)

Erklärung zur selbstständigen Bearbeitung einer Abschlussarbeit

Gemäß der Allgemeinen Prüfungs- und Studienordnung ist zusammen mit der Abschlussarbeit eine schriftliche Erklärung abzugeben, in der der Studierende bestätigt, dass die Abschlussarbeit „— bei einer Gruppenarbeit die entsprechend gekennzeichneten Teile der Arbeit [(§ 18 Abs. 1 APSO-TI-BM bzw. § 21 Abs. 1 APSO-INGI)] — ohne fremde Hilfe selbständig verfasst und nur die angegebenen Quellen und Hilfsmittel benutzt wurden. Wörtlich oder dem Sinn nach aus anderen Werken entnommene Stellen sind unter Angabe der Quellen kenntlich zu machen."

Quelle: § 16 Abs. 5 APSO-TI-BM bzw. § 15 Abs. 6 APSO-INGI

Erklärung zur selbstständigen Bearbeitung der Arbeit

Hiermit versichere ich,

Name: _____

Vorname: _____

dass ich die vorliegende Bachelorarbeit – bzw. bei einer Gruppenarbeit die entsprechend gekennzeichneten Teile der Arbeit – mit dem Thema:

Analyse und Visualisierung des Mehrwertes von neuen Features in Webanwendungen

ohne fremde Hilfe selbständig verfasst und nur die angegebenen Quellen und Hilfsmittel benutzt habe. Wörtlich oder dem Sinn nach aus anderen Werken entnommene Stellen sind unter Angabe der Quellen kenntlich gemacht.

Ort	Datum	Unterschrift im Original