HAW HAMBURG

# Bachelorarbeit

Mareile Beernink genannt Konjer

## Personalisierung im E-Commerce unter Verwendung von Realtime Streaming Analytics

*Fakultät Technik und Informatik*
*Department Informatik*

*Faculty of Computer Science and Engineering*
*Department Computer Science*

Mareile Beernink genannt Konjer

# Personalisierung im E-Commerce unter Verwendung von Realtime Streaming Analytics

Bachelorarbeit eingereicht im Rahmen der Bachelorprüfung
im Studiengang *Bachelor of Science Wirtschaftsinformatik*
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer: Prof. Dr. Olaf Zukunft
Zweitgutachter: Prof. Dr. Ulrike Steffens

Eingereicht am: 06. Mai 2021

**Mareile Beernink genannt Konjer**

**Thema der Arbeit**

Personalisierung im E-Commerce unter Verwendung von Realtime Streaming Analytics

**Stichworte**

E-Commerce, Personalisierung, Echtzeitanalyse, Event-Driven Architecture, Apache Flink, Ereignisorientierung, Big Data

**Kurzzusammenfassung**

Diese Arbeit befasst sich mit der Entwicklung eines Echtzeitsystems im Kontext von Personalisierung im E-Commerce. Hierzu wird unter der Berücksichtigung des Event-Driven Architecture Stils und mit Hilfe des Frameworks Apache Flink ein Prototyp entwickelt, der eine große Anzahl an Kundeninteraktionen verarbeiten und abhängig von verhaltensorientierten und soziodemografischen Merkmalen des Kunden personalisierte Angebote und Empfehlungen ausgeben kann. Bezüglich dessen werden spezifische funktionale sowie nicht funktionale Anforderungen an eine E-Commerce Plattform herausgearbeitet und anhand dieser die Konzeptionierung und Entwicklung aufgebaut.

**Title of Thesis**

Personalization in E-Commerce using Realtime Streaming Analytics

**Keywords**

e-commerce, personalisation, realtime streaming analytics, event-driven architecture, Apache Flink, event-driven processing, big data

**Abstract**

This thesis deals with the development of a real-time system in context of personalization in e-commerce. For this purpose, a prototype is developed that is able to process a large number of customer interactions and print personalized offers and recommendations depending on customer's behavior-oriented and socio-demographic characteristics. The prototype is considering the event-driven architecture style and is implemented with the help of Apache Flink. Specific functional and non-functional requirements for an e-commerce platform are analyzed and specify the base of the conception and development.

# Inhaltsverzeichnis

# Abbildungsverzeichnis

# Tabellenverzeichnis

# 1 Einleitung

Nicht erst seit der Corona Pandemie boomt der Online-Handel in Deutschland und der Welt. So wurden bereits 2019 in Deutschland im B2C-Bereich 59,2 Milliarden Euro durch E-Commerce umgesetzt [54]. Prognosen zufolge wird die Nutzerzahl von E-Commerce Plattformen weltweit bis 2024 auf über 4,5 Milliarden ansteigen [53]. Amazon, Pionier und einer der größten Marktakteure im Bereich Online Verkauf, zählte allein im Oktober 2020 ca. 543 Millionen Besuche auf ihrer Webseite [52].

Die große Anzahl an Nutzern birgt einerseits ein großes Potential für Unternehmen, andererseits müssen jedoch auch die verwendeten Systeme immer leistungsfähiger werden und stellen die Unternehmen somit vor Probleme. Denn nicht allein die Masse potentieller Kunden, sondern auch der steigende Konkurrenzkampf hat einen nicht zu vernachlässigenden Einfluss auf ihre Arbeit. Um der Konkurrenz Stand zu halten, gilt es, die Aufmerksamkeit des Kunden schnellstmöglich für sich zu gewinnen [49, S.145f.]. Dafür müssen Unternehmen die Interessen ihrer Kunden kennen und unmittelbar auf diese eingehen können.

Demzufolge ist die Beachtung von zwei Faktoren essentiell für den Erfolg eines E-Commerce Unternehmens. Zum einen müssen das Verhalten sowie die Vorlieben von Kunden bekannt bzw. prognostizierbar sein, um diese mit Hilfe von Angeboten und Empfehlungen möglichst individuell anzusprechen. Zum anderen ist es notwendig, den kontinuierlich eintreffenden Datenstrom von Kundeninteraktionen in Echtzeit zu verarbeiten und für die Analyse bereitzustellen [32, S.7f.].

An diesem Punkt setzt die Thesis an. Im Verlauf der Arbeit werden verschiedene Einflussfaktoren auf die Systeme herausgestellt und erläutert. Der Hauptteil der Thesis befasst sich mit der Analyse, Konzeption und Umsetzung eines Prototypen, der die Echtzeitverarbeitung von Streaming Daten und die Ausgabe von personalisierten Angeboten sowie Empfehlungen ermöglicht.

## 1.1 Problemstellung und Zielsetzung

Dadurch dass die Erwartungen von Kunden an E-Commerce Plattformen immer größer werden und zusätzlich der Wettbewerb im online Geschäft, u.a. auf Grund der niedrigen Einstiegshürden, stetig ansteigt, stehen Unternehmen vor dem Problem die Nutzer für sich zu gewinnen. Darüber hinaus steigt mit neuen Technologien die Menge an gesammelten Daten, die analysiert und zur Kundenansprache verwendet werden sollten.

Demzufolge gilt es als Zielsetzung dieser Thesis eine realitätsnahe Abbildung einer E-Commerce Plattform zu konzeptionieren und entwickeln, die einen Nutzer in Echtzeit mit geeigneten Angeboten und Empfehlungen auf Grundlage von vorhandenen Daten vom Webshop überzeugen kann.

## 1.2 Aufbau der Thesis

Zu Beginn der Thesis werden die theoretischen Grundlagen in Bezug auf den E-Commerce und die Echtzeitanalyse dargelegt. Die Grundlagen sollen dazu dienen, dem Leser einen umfassenden Einblick in die Thematik zu ermöglichen.

Im Anschluss an die Grundlagen werden in Kapitel 3 die essentiellen Eigenschaften und Gegebenheiten von Daten analysiert. Darüber hinaus werden in diesem Kapitel personalisierte Angebote und Empfehlungen behandelt. Abschließend werden anhand von Use Cases funktionale und nicht funktionale Anforderungen an den zu entwickelnden Prototypen aufgestellt.

Nachdem die Anforderungen definiert wurden, erfolgt in Kapitel 4 die Konzeptionierung des Prototypen und der damit verbundenen Auswahl von Systemkomponenten sowie der Festlegung der Business Rules. Im Anschluss folgt in Kapitel 5 die Beschreibung des festgelegten Realisierungsumfangs und ferner die Vorstellung ausgewählter Codesequenzen.

Zum Abschluss der Thesis werden unter Kapitel 6 die erzielten Ergebnisse ausgewertet sowie die gestellten Anforderungen auf ihre Erfüllung überprüft. Des Weiteren erfolgt eine kritische Betrachtung des entwickelten Prototypens, auf deren Basis die Realitätsnähe bewertet wird.

# 2 Grundlagen

Zu Beginn wird die notwendige Wissensbasis für das Verständnis der Arbeit vermittelt. Die zentralen Themengebiete werden hierfür unabhängig voneinander vorgestellt und im weiteren Verlauf der Arbeit zusammengeführt.

## 2.1 E-Commerce

Immer mehr Kunden entscheiden sich für einen Kauf im Internet [53]. Die sich den Kunden bietende Auswahl an Produkten und Dienstleistungen ist groß, Zeitpunkt und Standort der Bestellung sind flexibel wählbar und auf Kundenprobleme und -wünsche wird immer gezielter eingegangen. Doch nicht nur Kunden erkennen die Vorteile des E-Commerce, auch Händler nutzen die sich ihnen bietenden Möglichkeiten, Kunden in aller Welt zu gewinnen [32, S.3ff.]. Der dadurch entstehende verschärfte Wettbewerb führt dazu, dass die Geschäftsmodelle konsequent auf den Kunden ausgerichtet werden müssen [34, S.7].

In den folgenden Kapiteln werden die Grundlagen zum E-Commerce, die Eingrenzung auf Personalisierung und das Thema Datenschutz dargelegt. Die Kapitel Empfehlungssysteme und Kunden-Clusterung schließen die Grundlagen zum E-Commerce ab.

### 2.1.1 Einblick in das E-Commerce

#### 2.1.1.1 Definition E-Commerce

„Im Kern geht es [beim E-Commerce] um den elektronischen Handel mit Waren und Dienstleistungen, deren Transaktion, d. h. die Anbahnung, der Abschluss und die Abwicklung des Kaufs oder Verkaufs, über das Internet mithilfe interaktiver Informations-

und Kommunikationstechnologien durchgeführt wird.“ [26, S.2] Unter dem Begriff E-Commerce sind demnach auf elektronischem Weg getätigte wirtschaftliche Tätigkeiten zur Umsatzgenerierung zu verstehen. Der Handel mit Waren und Dienstleistungen kann dabei in die vier Marktformen Business to Consumer (B2C), Business to Business (B2B), Consumer to Consumer (C2C), Consumer to Business (C2B) kategorisiert werden [26, S.40ff.].

In Abbildung 2.1 wird die Einbettung des E-Commerce in das E-Business ersichtlich. In diesem hat es sich als fester Teilbereich etabliert [26, S.4].

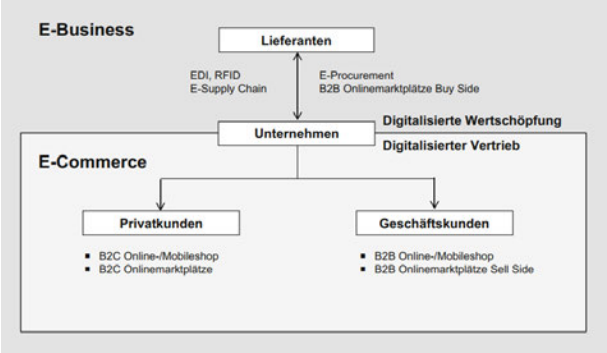



Abbildung 2.1: Abgrenzung E-Business und E-Commerce [26, S.5]

„E-Business steht für die elektronische Unterstützung, Abwicklung und Aufrechterhaltung von Geschäftsprozessen als Leistungsaustauschprozesse zwischen Unternehmen auf Basis computergestützter Netzwerke der Internettechnologie.“ [60] Das E-Business befasst sich demnach sowohl mit unternehmensinternen als auch mit externen und übergreifenden Geschäftsprozessen, während sich E-Commerce mit dem Vertrieb, also den Handelstransaktionen zwischen Geschäftspartnern, über elektronische Wege befasst [2, S.36f.].

#### 2.1.1.2 Entwicklungsgeschichte

Mit den Gründungen von Amazon (1994) und eBay (1995) sowie verschiedenen Suchmaschinen (u.a. Yahoo) etablierte sich der Begriff E-Commerce. In der Folge rückten Geschäftsmodelle, die die Besonderheiten des E-Commerce und betriebswirtschaftliche Grundsätze miteinander verbanden, in den Mittelpunkt.

Abbildung 2.2 zeigt die Entwicklungsgeschichte des E-Commerce nach Engelhardt und Magerhans. Die derzeitige Phase wird als „Web der Daten“ bezeichnet und hat erst vor ein paar Jahren begonnen. Die unmittelbaren Entwicklungen zeigen, dass das Sammeln von Daten immer entscheidender für den Erfolg von Geschäftsmodellen wird. Mit dieser Entwicklung in Verbindung stehende Begriffe sind u.a.: Internet der Dinge, Big Data, Data Mining und Analytics. Die Grundlage von Geschäftsmodellen im E-Commerce bildet das Sammeln von großen Datenmengen in Echtzeit [27, S.7ff.] [26, S.6f.].

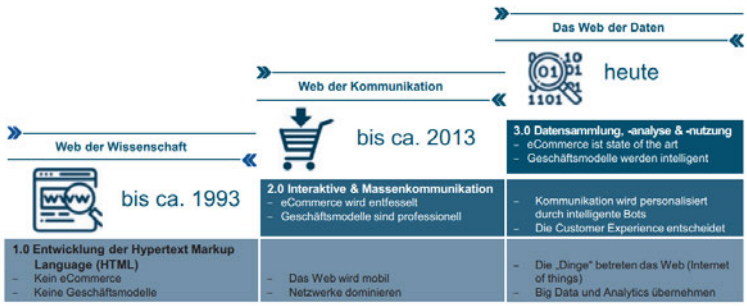



Abbildung 2.2: Entwicklungsgeschichte des E-Commerce [27, S.8]

#### 2.1.1.3 Zentrale Erfolgsfaktoren des E-Commerce

Während sich Unternehmen früher über mehrere Generationen etabliert haben und kontinuierlich gewachsen sind, sind heutige Unternehmen durch ihren rasanten Einstieg geprägt und können innerhalb weniger Jahre eine beherrschende Stellung am Markt einnehmen [26, S.7]. Beispiele für schnell wachsende Unternehmen sind Google, Apple, Facebook und Amazon (auch bekannt als GAFA).

Unter zwei weiterten Erfolgsfaktoren bezeichnet Große Holtforth [32, S.6ff.] das Data Driven Marketing sowie die Customer Centricity als die Grundlage für den großen Erfolg und das schnelle Wachstum der GAFA Unternehmen. Mit Hilfe des **Data Driven Marketing** ist es möglich das Kundenverhalten wesentlich detaillierter nachzuvollziehen und die gesammelten Kundendaten zur weiteren Produktentwicklung und Kundenansprache zu nutzen. Anders als traditionelle Unternehmen, bei denen das Produkt im Mittelpunkt steht, nutzt das **Customer Centricity** die Erfahrungen mit den Kunden als Grundlage für die Weiterentwicklung von Leistungen. Das Ziel ist es, sich mit den Kunden zu vernetzen und auf diese Weise die Nutzererfahrungen zu maximieren.

#### 2.1.1.4 Chancen und Risiken des E-Commerce

Durch das E-Commerce haben sich viele neue Möglichkeiten sowohl für Unternehmen als auch für Kunden gebildet [26, S.30ff.]. In der folgenden Tabelle 2.1 werden einzelne der vielen Chancen und Risiken für beide Perspektiven dargelegt:

| | Anbieter | Kunden |
|---|---|---|
| Chancen | - Produkte und Dienstleistungen können unabhängig von Ort und Zeit angeboten werden<br>- Durch fehlende räumliche Einschränkungen kann das Sortiment erweitert werden<br>- Kundenprofile ermöglichen personalisierte Angebote und Services | - Produkte und Dienstleistungen können unabhängig von Ort und Zeit gekauft werden<br>- Reduzierung von Zeitaufwand und emotionaler Mühe<br>- Personalisierte Angebote und Vorschläge helfen dem Kunden bei seiner Entscheidung |
| Risiken | - Intensivierung des Wettbewerbs durch eine gestiegene Preistransparenz und Vergleichbarkeit<br>- Steigende Kosten durch hohe Retourenquoten, Pflege des Onlineshops und der Integration in die IT-Infrastruktur<br>- Entstehende Kosten durch IT-Security zum Schutz von personenbezogenen Daten | - Die Anzahl an Angeboten und Produkten kann zu einer Überforderung der Kunden führen<br>- Produkte und Dienstleistungen entsprechen nicht den Erwartungen<br>- Entstehende Risken bei Zahlungsvorgängen und bei dem Umgang mit persönlichen Daten |

Tabelle 2.1: Chancen und Risiken für Anbieter und Kunden

### 2.1.2 Personalisierung

Durch die große Auswahl an Angeboten im Internet, nimmt die Personalisierung im Webshop eine immer größere Rolle ein. Das liegt u.a. daran, dass das überwältigende Angebot zur Informationsüberlastung beim Kunden führen kann. Die Personalisierung soll die

Suche und das Auffinden von relevanten Produkten erleichtern und die Entscheidungsfindung unterstützen. Für Unternehmen bietet sie durch die Differenzierungsmöglichkeit einen entscheidenden Wettbewerbsvorteil [56, S.2ff.].

#### 2.1.2.1 Definition Personalisierung

Stüber [56, S.12] erklärt den Begriff Personalisierung als die individuelle Darbietung von angepassten Dienstleistungen an Konsumenten. Diese äußern sich zum einen in einem angepassten zwischenmenschlichen Verhalten (z. B. namentliche Ansprache, Small Talk) und zum anderen in einer angepassten Produktdarbietung (z. B. Kaufempfehlungen).

Nach der Definition von Stüber reagiert ein Webshop somit individuell auf seine Kunden. Durch einen persönlichen Account kennt die Webseite den Namen des Kunden und kann diesen persönlich ansprechen. Je weniger Aktionen ein Kunde durchführen muss, um an sein Ziel zu kommen, desto positiver ist seine Wahrnehmung. Demnach sollte ein Ziel der Personalisierung sein, zu gewährleisten, dass der Kunde mit minimalem Aufwand sein Ziel erreichen kann [18, S.48ff.].

Zur Bezeichnung der Personalisierung lassen sich in der Literatur unterschiedliche Begriffe finden. Beispielsweise teilt die **Segmentierung** heterogene Gruppen hinsichtlich bestimmter Eigenschaften in homogene Gruppen auf [56, S.11]. Die **Individualisierung** wiederum adressiert immer nur einen einzelnen Kunden. Sie kann durch Unternehmen initiiert oder durch den Kunden selbst durchgeführt werden. Der Anbieter personalisiert seine Kunden, während der Kunde mittels **Customization** einzelne Elemente anpassen kann. Die Personalisierung kann sich also sowohl auf einzelne Kunden als auch auf eine Kundengruppe mit ähnlichen Persönlichkeitseigenschaften beziehen. Die zuvor aufgegriffene Definition der Personalisierung konzentrierte sich lediglich auf die individuelle Personalisierung.

Nachdem nun verdeutlicht wurde, für wen die Personalisierung gilt (Zielgruppe) und definiert wurde, wer die Personalisierung durchführt (Initiator), stellt sich noch die Frage: Was personalisiert werden kann? (Objekt). Hierzu werden vier mögliche Objekte genannt. Personalisierbar sind die **Informationen/ Inhalte** und ihre **Darstellung**, die **Auswahl des Kanals**, auf dem die Inhalte bereitgestellt werden sollen, und die **Art und Weise** wie die Interaktion mit den Inhalten stattfinden soll [14, S.27ff.].

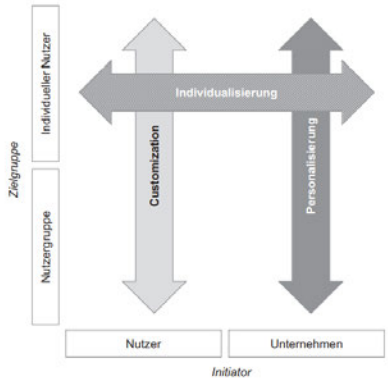



Abbildung 2.3: Abgrenzung Personalisierung, Individualisierung und Customization [14, S.28]

#### 2.1.2.2 Formen der Personalisierung

In der Literatur werden drei Formen der Personalisierung definiert: personalisierte Empfehlungen, personalisierte Ansprache sowie personalisierte Interfaces [14, S.29ff.].

Unter **personalisierten Empfehlungen** werden Systeme verstanden, die aus einer Menge von Produkten die Produkte ermitteln, die die Präferenzen des Kunden bestmöglich abdecken. Durch sie kann die Informationsüberlastung verringert werden, im Idealfall enthalten die Empfehlungen nur noch die für den Kunden relevanten Produkte.

Eine weitere Form ist die **personalisierte Ansprache**. Hierzu verwendet die Webseite die persönlichen Daten des Kunden und begrüßt diese bspw. mit Namen. Für gewöhnlich erzeugt dieses Vorgehen eine gesteigerte Aufmerksamkeit. Allerdings kann das Nutzen der persönlichen Daten auch zu Datenschutz-Bedenken führen, die in Kapitel 2.1.3 eingehend erläutert werden.

Durch **personalisierte Interfaces** können einzelne Webseiten-Elemente in verschiedenen Varianten erstellt werden, sodass jeder Kunde die Variante angezeigt bekommt, die ihn persönlich am besten anspricht. Dadurch kann die User Experience verbessert und der Entscheidungsprozess verkürzt werden. So drehte Netflix bspw. für die Serie „House of Cards“ mehrere Trailer und erstellt für Filme unterschiedliche Vorschaubilder. Um das Interesse der Kunden zu wecken, werden die erstellten Trailer und Vorschaubilder passend zum jeweiligen Kundentyp gezeigt.

#### 2.1.2.3 Kundenvorteile sowie Relevanztreiber

Die Personalisierung im Onlinehandel unterstützt die Kunden, indem sie die Komplexität reduziert und die Informationsüberlastung mindert. Dadurch kann der Entscheidungsprozess verkürzt und die Effektivität erhöht werden. Langes Suchen nach einem Produkt kann reduziert und der Kundenservice verbessert werden. Durch eine transparente Datenverarbeitung und -kontrolle kann Vertrauen gewonnen werden. Letztendlich erhöht die Personalisierung die Kundenzufriedenheit und steigert damit die Kundenbindung [56, S.20].

Allerdings steigt durch die zunehmende Vielfalt digitaler Angebote, Anbieter und Endgeräte auch der Wettbewerb um die Aufmerksamkeit eines Kunden. Die Zeit, um den Kunden von einem Angebot zu überzeugen, wird demnach immer kürzer. Neben dem Relevanztreiber Zeit nimmt auch die Screen Größe des Kunden ab. Während die Kunden früher primär über einen Computer ihre Ware bestellt haben, lassen sich heute viele Kunden mit Hilfe ihres Smartphones durch die Websites bzw. Apps treiben. Des Weiteren muss beachtet werden, dass die Endgeräte immer weiter in den persönlichen Raum eintreten.

Auf Grund dieser drei Faktoren (Zeit, Fläche, Nähe) ist davon auszugehen, dass die Kunden zukünftig mehr Personalisierung einfordern werden. Um Kunden nicht mit der zunehmenden Vielfalt von Produkten zu überfordern, ist es notwendig die Aufmerksamkeit möglichst schnell auf die für sie relevanten Produkte zu lenken [49, S.145ff.].

#### 2.1.2.4 Personalisierte Preise

Personalisierte Preise verfolgen das Ziel, die Preisbereitschaft des Kunden vollständig auszureizen. Das heißt Kunden, deren Preisbereitschaft unterhalb des Ursprungpreises liegt, erhalten individuelle Preisnachlässe, während Kunden, die einem wenig preissensiblen Segment zugeordnet sind, keine Nachlässe erhalten [48, S.71ff.].

2015 hat das LINK Institut für Markt und Sozialforschung GmbH [37] eine Umfrage zum Bewusstsein von Verbrauchern zur Preisdiskriminierung im Onlinehandel durchgeführt. Die Umfrageergebnisse zeigen, dass die Erfahrungen mit Preisdiskriminierung in der Regel sehr gering sind. Angebote über Emails werden dagegen vom Großteil der Befragten wahrgenommen. Jeder Vierte hat zudem schon die Erfahrung gemacht, dass ihm nach der Registrierung ein niedrigerer Preis angeboten wird. Preisdifferenzierungsmaßnahmen

werden von fast allen Befragten als unfair empfunden. Allerdings werden transparente Transaktionen, wie die vom Buchungszeitpunkt abhängigen Preise in der Tourismusbranche, bereits von der Bevölkerung akzeptiert.

#### 2.1.2.5 Personalisierung in der Praxis

Im Folgenden wird exemplarisch aufgeführt, wie und mit welchen Mitteln Unternehmen ihre Kanäle personalisieren. Der Onlineshop von OTTO [46] verwendet bspw. Shopteaser auf der Homepage und auf Sortimentseinstiegseiten. Shopteaser sind für die Navigation auf tiefere Ebenen zuständig. Zudem werden verkaufsfördernde Maßnahmen, wie eine Befreiung von Versandkosten oder Rabatte auf bestimmte Sortimente, eingesetzt. Sie versuchen die Kaufentscheidung positiv zu beeinflussen, indem sie die jeweiligen Kaufwahrscheinlichkeiten in Echtzeit prognostizieren und so gezielte Maßnahmen einsetzen können [49, S.156ff.].

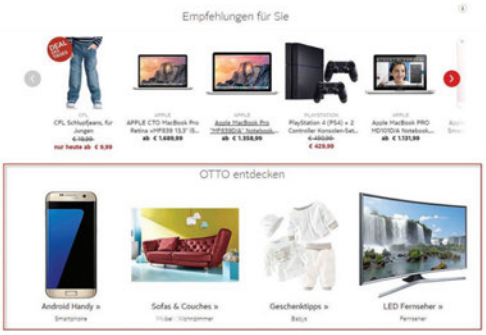



Abbildung 2.4: Shopteaser auf Otto.de [49, S.157]

Der Online Fashionshop About You [1] hat sich bei der Entwicklung von Personalisierungsmaßnahmen von Social-Media-Kanälen inspirieren lassen. Bei About You können Kunden Marken, Stars und Looks folgen und ihre Benutzerprofile stehen im Fokus. Lieblingsfarben, Konfektionsgrößen, Marken etc. können im Profil gespeichert werden und führen dazu, dass den Kunden relevante Produkte angezeigt werden [20, S.166f.].

#### 2.1.2.6 Mehrwert der Personalisierung

Ein Mehrwert kann aus der Personalisierung nur gewonnen werden, wenn der Kunde diese positiv wahrnimmt. So zeigen Studien und Experimente, dass z.B. Wiederbesuchs- und Kaufabsichten durch Personalisierung gesteigert werden können [14, S.11ff.].

Es wird u.a. gezeigt, dass die psychografischen Merkmale (wie Motivation, Meinungen, Wünsche, Werte, Lebensstil) aus der Klick- und Kaufhistorie ermittelt werden können und demnach das optische Erscheinungsbild der Webseite (Text-Bild-Verhältnis) verändert werden kann, welches wiederum den Komplexitätsgrad beeinflusst. Des Weiteren können Informationen über den individuellen Lebensstil eines Kunden aus seinen Kundenkarteninformationen extrahiert werden und somit ebenso das Erscheinungsbild der Webseite beeinflussen. In den meisten Studien wird jedoch auf eine Betrachtung der psychografischen Daten verzichtet. Stattdessen werden Klick- und Kaufhistorie oder Informationen, die bei der Registrierung entstehen, zur Personalisierung verwendet. Untersuchungen zeigen, dass Empfehlungen für Produkte, die eine hohe Schnittmenge mit anderen bereits betrachteten oder gekauften Produkten haben, eine höhere Klickrate haben als nicht personalisierte Empfehlungen.

Jedoch zeigen die Studien auch, dass die Verwendung von personalisierten Empfehlungen nur bei vertrauenswürdigen Webshops zu einer Erhöhung der Klickrate führen. Hat der Kunde das Gefühl einen Kontrollverlust über seine Daten zu erhalten, werden personalisierte Anzeigen seltener angeklickt. Entscheidend für die Generierung hoher Klickraten ist somit, dass die Datensammlung möglichst transparent kommuniziert wird und so Datenschutzbedenken minimiert werden.

### 2.1.3 Datenschutz

Auf Grund der Personalisierung werden personenbezogene Daten immer wichtiger und essenzieller für den Erfolg des Onlinehandels und anderen Bereichen. Persönliche und sensible Daten werden in großer Menge gesammelt, angereichert und gespeichert. Auf diese Weise lassen sich Kunden in kurzer Zeit analysieren und mit gezielter Werbung ansprechen [59, S.238].

Die kundenseitige Akzeptanz der Datennutzung hängt dabei stark von den empfundenen Vorteilen ab. Während bei personalisierter Werbung wie z.B. „andere Kunden kauften auch“ bei Amazon oder bei passenden Filmempfehlungen auf Netflix nicht hinterfragt

wird, woher die Unternehmen die eigenen Präferenzen kennen, reagieren Kunden irritiert und verärgert, wenn sie dasselbe Produkt wiederholt auf unterschiedlichen Seiten angezeigt bekommen [28, S.293]. Um Irritationen zu beseitigen und primär eine vertrauensvolle Interaktion zwischen Unternehmen und Kunden im Internet herzustellen, muss die Privatsphäre des Nutzers geschützt werden.

Neben angepassten Verordnungen von bereits existierenden Gesetzen wie bspw. dem Bürgerlichen Gesetzbuch (BGB), dem Urheberrecht (UrhG) oder dem Markengesetz (MarkenG), beschäftigt sich insbesondere die 2018 in Kraft getretene Datenschutz-Grundverordnung (DSGVO) der Europäischen Union (EU) mit der Verarbeitung von personenbezogenen Daten [26, S.19ff.].

### 2.1.4 Empfehlungssysteme

Empfehlungssysteme tragen einen essenziellen Anteil an der Personalisierung von E-Commerce Plattformen. Durch sie können dem Kunden potenziell interessante Produkte empfohlen werden, Entscheidungen der Konsumenten erleichtert und die unterschiedlichen Charaktere individuell angesprochen werden [56, S.4].

#### 2.1.4.1 Definition Empfehlungssystem

Empfehlungssysteme analysieren die Kunden, Produkte und die zu Grunde gelegte Situation auf Ähnlichkeiten, um relevante und individuelle Produktempfehlungen auszusprechen. Sie nutzen für ihre Analyse sowohl gespeicherte Daten ihrer Kunden als auch aktuelles Kauf- und Klickverhalten. Zusätzlich berücksichtigen sie die vorhandenen Produkte und ihre Analogien. Aus Unternehmensperspektive können auf diese Weise Kaufempfehlungen für Zusatzprodukte (Cross-Selling) oder für alternativ Produkte (Up-Selling) entwickelt werden [56, S.27f.].

Des Weiteren können die Interessen der Kunden optimal angesprochen und somit die Benutzererfahrungen im Webshop verbessert werden. Aus Kundensicht senken Empfehlungssysteme die anfallenden Informationskosten und können so zu vermehrten Käufen führen [56, S.37]. Schlussendlich sollten Empfehlungssysteme die Kundenbindung stärken und höhere Verkaufszahlen erzielen.

#### 2.1.4.2 Arten von Empfehlungssystemen

Empfehlungssysteme lassen sich zunächst in nicht personalisierte (allen Kunden werden identische Empfehlungen ausgesprochen) und personalisierte Empfehlungssysteme unterschieden. In dieser Arbeit werden nur die personalisierten Systeme betrachtet und in verschiedene Ansätze unterteilt. Die drei erprobten personalisierten Empfehlungstechniken sind das Collaborative-, das Content-Based-Filtering-Verfahren sowie hybride Formen. Für alle drei Systeme sind aussagekräftige und große Datenbasen zwingend notwendige Voraussetzungen.

**Collaboratives-Filtering-Verfahren**
Beim Collaborativen-Filtering erhalten die Nutzer ihre Produktempfehlungen auf Basis des Verhaltens anderer Nutzer mit ähnlichem Profil. Benutzerprofile geben u.a. darüber Auskunft, welche empfohlenen Produkte die Aufmerksamkeit des Kunden erregt haben. Ein wesentlicher Vorteil ist, dass sie für alle Arten von Produkten verwendet werden können.

**Content-Based-Filtering-Verfahren**
Beim Content-Based-Filtering steht nicht die Ähnlichkeit von Benutzerprofilen im Mittelpunkt, sondern die Verwandtschaft von Empfehlungselementen. Ausgehend vom Nutzer wird analysiert welche Produkte zu seinem Profil passen. Die Analyse erfolgt auf Basis der Produkte, die der Benutzer in der Vergangenheit als gut bewertet hat. Der hohe Initialaufwand und die kontinuierliche Pflege der Produktbeschreibungen zeichnen die Nachteile dieses Verfahrens aus. Die Vorteile des Verfahrens liegen darin, dass keine Mindestanzahl an Nutzern notwendig ist, die Empfehlungen inhaltsbezogen umgesetzt werden können und diese für den Nutzer transparent und steuerbar sind.

**Hybride Empfehlungstechniken**
Hybride Empfehlungssysteme kombinieren verschiedene Empfehlungstechniken und sind heute weitgehend in der Praxis zu finden. Hybride Empfehlungstechniken berücksichtigen sowohl das Verhalten eines einzelnen Nutzers als auch das aller anderen Nutzer. Sie verfolgen das Ziel grundlegende Probleme zu eliminieren und die Vorteile der verschiedenen Techniken auszuschöpfen. Die Collaborativen Empfehlungssysteme werden um die inhaltsbasierten Profile der Nutzer erweitert. So können sowohl die individuellen Kundeninteressen als auch die Gemeinsamkeiten zwischen den Produkteigenschaften erkannt werden [36, S.265-271].

### 2.1.5 Kunden-Clusterung

Eine weitere Methode, um zielgruppengerechte Angebote und Empfehlungen ausgeben zu können ist das Clustern von Kunden.

#### 2.1.5.1 Definition Clustering

Beim Clustering werden Daten in möglichst homogene Gruppen unterteilt. Beim klassischen Clusterverfahren werden Daten festen Gruppen zu geordnet, d.h. die Zuordnung ist eindeutig. Die Clusteranalyse gehört zu der Kategorie des unüberwachten Lernens, da noch nicht bekannt ist, nach welchen Mustern gesucht wird bzw. welche Kunden zueinander passen [44, S.8ff.].

#### 2.1.5.2 Verfahren der Clusteranalyse

Zur Durchführung einer Clusteranalyse stehen verschiedene Verfahren zur Auswahl, die sich in der Wahl des Proximitätsmaßes und der Wahl des Gruppierungsverfahrens unterscheiden. Das **Proximitätsmaß** misst Ähnlichkeiten bzw. Unähnlichkeiten zwischen Objekten. Das **Gruppierungsverfahren** legt fest, nach welcher Vorgehensweise das Zusammenfassen von ähnlichen Objekten zu Gruppen erfolgen soll [15, S.443]. Grundlegend wird zwischen den hierarchischen und den partitionierenden Verfahren unterschieden.

Innerhalb der **Hierarchischen Verfahren** wird eine weitere Unterscheidung in divisive und agglomerative Verfahren getroffen. Während divisive Verfahren die Menge aller Objekte sukzessive unterteilen (beim Start befinden sich alle Objekte in einer Gruppe), werden beim agglomerativen Verfahren Objekte sukzessive zu Clustern zusammengefasst (beim Start befindet sich jedes Objekt in seiner eigenen Gruppe).

Beim **partitionierten Verfahren** wird die Menge aller Objekte zunächst zufällig in Cluster aufgeteilt. In den folgenden Schritten werden die einzelnen Objekte so lange in andere Cluster verschoben bis homogene Cluster entstanden sind [57, S.49f.].

#### 2.1.5.3 Kriterien zur Bildung von Kundenclustern

Das Segmentieren der Kunden soll Auskunft über die verschiedenen Kundentypen sowie ihre Erwartungen und Einkaufsgewohnheiten geben, um diese gezielter ansprechen zu können. Wie Zaharia und Hackstetter [61, S.46f.] sowie Glaser [31, S.149f.] beschreiben, können Kunden anhand ihres Verhaltens sowie ihrer soziodemografischen und psychografischen Merkmale in verschiedene Cluster unterteilt werden.

Zu dem **Verhalten** des Kunden zählen Kaufentscheidungen wie z.B. die Preisklassen in der der Kunde einkauft oder ob Wert auf Rabattaktionen gelegt wird, das Klickverhalten wie z.B. an welchen Sortimenten und Marken Interesse gezeigt wird oder aber welches Device und welchen Kanal der Kunde verwendet.

Zu den **Soziodemografischen Merkmalen** zählen u.a. das Alter, Geschlecht und der Wohnort des Kunden. Auch können das Bildungsniveau, der ausgeführte Beruf oder das Einkommen Berücksichtigung finden.

**Psychografische Merkmale** sind am schwersten zu bestimmen. Beispielsweise zählen die Kaufabsicht - also welchen Nutzen der Kunde beim Einkaufen verfolgt – oder das Shoppingerlebnis zu den psychografischen Merkmalen.

#### 2.1.5.4 Beispielhaftes Kundencluster

Bernhard und Mühling [18, S.51f.] unterteilen Kunden beispielhaft in drei Typen: den Jäger, den Sucher und den Sammler.

Wenn ein **Jäger** einen Webshop besucht, hat er eine feste Vorstellung von dem, was er sucht. Die Suche erfolgt zielgerichtet nach seiner Vorstellung und falls er sein passendes Produkt findet, wird dieses unmittelbar erstanden. Findet er das gesuchte Produkt nicht, verlässt er den Shop und sucht auf einer anderen Website weiter.

Ein Kunde des Typs **Sucher** weiß vage, was er will. Er sucht bspw. nach einem Smartphone, weiß aber noch nicht, welches Modell und welchen Hersteller er aussuchen wird. Bei seiner Suche grenzt er durch seine festgelegten Rahmenbedingungen ein (wie z.B. den Preis des Produktes) und vergleicht verschiedene Produkte. Hat der Kunde sich entschieden, kauft er sein Produkt und zeigt oft Interesse am Kauf von weiterem Zubehör.

Ein **Sammler** hat dagegen keine Vorstellung oder Erwartungen. Er stöbert durch die Ware im Shop und lässt sich inspirieren.

Ein Kunde lässt sich in der Praxis jedoch selten nur einem Typen zuordnen.

## 2.2 Echtzeitanalyse

Um im ausschlaggebenden Moment die richtigen Entscheidungen treffen zu können, ist es häufig notwendig, Analysen in Echtzeit durchführen zu können. Zusammenhänge, Abhängigkeiten und Korrelationen automatisiert zu erkennen sind notwendige Voraussetzungen, um Datenströme zeitnah verarbeiten zu können [22, S.2].

Dieses Kapitel beschäftigt sich mit den Grundlagen und der Begriffserläuterung von Echtzeitanalyse, stellt die Event-Driven Architecture im Zusammenhang mit dem Complex Event Processing vor und ermöglicht einen Überblick über die typischen Anwendungsfelder von ereignisgesteuerten Systemen.

### 2.2.1 Kernelemente der Echtzeitanalyse

#### 2.2.1.1 Big Data

Die zur Verfügung stehenden Daten bilden die Grundlage jeder Analyse. Wie in der Einleitung dieser Thesis beschrieben, steigen die Nutzerzahlen im E-Commerce zunehmend an und damit auch das entstehende Datenvolumen. Große Speicherkapazitäten, digitalisierte Produkte sowie eine große Vernetzung ermöglichen das Generieren und Speichern von großen Datensätzen.

Unter Big Data werden große, heterogene Datenmengen mit hohen Verarbeitungsgeschwindigkeiten verstanden. Die amerikanische Marktforschungsagentur Gartner definiert Big Data wie folgt: „Big data is high-volume, high-velocity and/or high-variety information assets that demand cost-effective, innovative forms of information processing that enable enhanced insight, decision making, and process automation.“ [30]

Die in der Definition genannten V-Eigenschaften Volume (umfangreicher Datenbestand), Velocity (hohe Verarbeitungsgeschwindigkeit) und Variety (Vielfalt von Datenformaten) werden häufig genutzt, um den Begriff Big Data zu erläutern. Zudem fallen häufig zwei

weitere Eigenschaften: Value (Unternehmenswert) und Veracity (Wahrhaftigkeit) [40, S.4ff.].

**Herausforderungen von Big Data**
Um mit Hilfe von Big Data Vorteile im Wettbewerb zu erreichen, müssen sich Unternehmen den damit verbundenen Herausforderungen stellen.

Als Basis muss eine entsprechende technische Ausstattung eingerichtet werden. Dabei besteht die Schwierigkeit, die heterogenen Formate (strukturierte, unstrukturierte und semistrukturierte Daten) mit einer hohen Verarbeitungsgeschwindigkeit speichern und verarbeiten zu können.

Weitere große Herausforderungen stellen das Personal und die Aufbereitung der Analysen für unterschiedliche Interessensgruppen in einem Unternehmen dar. Während sich die IT-Abteilung um die Hard- und Software für das Big-Data-System kümmert, kann es eine Analyseabteilung mit besonderen Kenntnissen und Analyse-Skills geben. Auch die einzelnen Fachabteilungen benötigen Informationen um ihre Arbeit datenfundiert auszuführen. Für die Analyseabteilungen werden entsprechende Experten benötigt, gleichzeitig müssen auch in anderen Abteilungen Big-Data Kompetenzen aufgebaut werden.

Zudem darf der Datenschutz und der damit verbundene verantwortungsvolle Umgang mit Daten als Komplikation bei der Speicherung und Verarbeitung von Daten nicht unterschätzt werden [35, S.12-20].

Bei den zahlreichen Herausforderungen ist es nicht überraschend, dass Business Entscheidungen noch sehr eingeschränkt auf Big-Data-Analysen beruhen. Die Mittelstandsbefragung der Commerzbank „Rohstoff des 21. Jahrhunderts: Big Data, Smart Data – Lost Data?“ [4] aus dem Jahr 2018 zeigt, dass bei mittelständigen Unternehmen lediglich 8 Prozent der Unternehmen die Nutzung und Analyse von Big Data gezielt durchführen. Dennoch sehen 81% Prozent der Befragten die Chancen und die existenziellen Notwendigkeiten im Zusammenhang mit Big Data.

#### 2.2.1.2 Datenstrom

In einem Datenstrom fließen kontinuierlich digitale Daten, die in der Regel zeitlich geordnet werden können. Die Datenmenge pro Zeiteinheit kann dabei hingegen sehr unterschiedlich sein [29, S.376]. Datenströme spiegeln reelle Ereignisse wider. Dabei beschreiben die einzelnen Ereignisse zumeist wenig komplexe Situationen und können mit

wenigen Daten beschrieben werden. Ein einzelner Datensatz, der z.B. durch einen Kunden beim Anklicken eines Werbebanners im Webshop erstellt wird, drückt ein Ereignis aus.

Ereignisse treten in der Praxis sehr häufig auf. Außerdem laufen die Datenströme kontinuierlich ohne Begrenzung fort und führen dazu, dass die Datenmenge so groß wird, dass eine dauerhafte Speicherung der gesamten Daten nicht möglich ist. Zusätzlich können Datensätze der Ströme miteinander in Beziehung stehen.

Datenströme werden dadurch charakterisiert, dass sie Live-Daten mit impliziten Beziehungen zueinander beinhalten, sie feingranular, hochfrequent und aus diesem Grund nur volatil speicherbar sind [22, S.2].

#### 2.2.1.3 Ereignis

„Ein Ereignis kann alles sein, was passiert oder von dem erwartet wird, dass es passiert [...]. Im Allgemeinen bezieht sich ein Ereignis auf die Veränderung eines Zustands [...]. “ [22, S.9] Geschäftsprozesse in Unternehmen werden durchgehend von erheblichen Mengen interner und externer Ereignisse gesteuert. Ereignisse können Chancen für den Prozess bieten oder unwichtig für weitere Entscheidungen sein. Hierbei wird in erwartete und unerwartete Ereignisse unterschieden [21, S.5ff.].

Eine nicht abbrechende Folge von einfachen Ereignissen wird als **Ereignisstrom** bezeichnet. Innerhalb eines Ereignisstroms können unterschiedliche Ereignistypen vorkommen. Ein Ereignistyp weist besondere Attribute auf, die neben den notwendigen Informationen (Typ, Zeitstempel und Ereignis ID), das Ereignisobjekt definieren. Resultieren die Objekte zudem aus unterschiedlichen Quellen, kann die Herkunft des Ereignisses als weitere Information festgehalten werden [33, S.27f.].

Sowohl die Menge als auch die Vielfalt von Ereignissen steigt kontinuierlich an. Die Gründe hierfür liegen in der Kommunikation mit großen Sensornetzwerken und dem stetig wachsenden Datenfluss. Aber auch die zunehmende fachliche sowie technische Komplexität tragen ihren Teil zum vielfältigen Wachstum von Ereignissen bei. Unternehmen entlang des Wertschöpfungsprozesses sind aufeinander abgestimmt und Prozesse aneinandergekoppelt. Die heterogene IT-Infrastruktur, bestehend aus alten und neuen Techniken, reicht von den unterschiedlichen Hardwarekomponenten über verschiedene Programmiersprachen bis hin zu heterogenen Datenformaten [21, S.15f.].

#### 2.2.1.4 Ereignismuster und Abstraktionsebenen

Um aus dem ansteigenden Ereignisfluss wertvolle Informationen gewinnen zu können, müssen komplexe Muster zwischen Ereignissen erkannt, Ereignisse auf unterschiedlichen Abstraktionsebenen betrachtet, kausale Beziehungen verstanden und Aktionen geschlussfolgert werden.

Zusammenhänge, Abhängigkeiten und Korrelationen zwischen Ereignissen bilden spezifische Muster. Da die Ereignisse, die in Beziehung zueinanderstehen, jedoch weder selten direkt aufeinander folgen noch am gleichen Ort auftreten, müssen die Muster über einen längeren Zeitraum gesucht werden [21, S.18f.].

Wird ein Muster erkannt, ergeben die zusammenhängenden, einfachen Ereignisse ein komplexes Ereignis auf einer höheren Abstraktionsebene. In der Literatur findet eine Unterscheidung zwischen einfachen und komplexen Ereignismustern statt. Zu den einfachen Ereignismustern zählen Einzelereignisse oder boolesche Kombinationen. Bei komplexen Ereignismustern spielen zudem auch die Reihenfolge und Zeitfenster für das Auftreten der Ereignisse eine Rolle [22, S.3ff.]. Während technische Ereignisse wie z.B. die Registration eines http-Request einen niedrigen Abstraktionsgrad besitzen, befinden sich Geschäftsereignisse, die einen fachlichen Sachverhalt beinhalten wie z.B. die Annullierung einer Bestellung durch einen Kunden, auf einer höheren Ebene [21, S.47ff.]. Auf höheren Abstraktionsebenen befinden sich demnach die aussagekräftigeren Ereignisse.

Um den Grund für das Auftreten eines Sachverhaltes verstehen zu können, ist es entscheidend kausale Abhängigkeiten zwischen Ereignissen nachvollziehen zu können. Ein kausaler Zusammenhang besteht zwischen einfachen Ereignissen, die durch eine Mustererkennung gruppiert wurden, und dem daraus gebildeten komplexen Ereignis. Nur wenn nach der Identifikation eines Musters auch eine fachliche Konsequenz abgeleitet wird, zeigt sie Wirkung. Fachliche Konsequenzen können z.B. die Erzeugung eines Ereignisses auf einer höheren Abstraktionsebene oder die Ausführung einer fachlichen Aktion bedeuten [21, S.18-22].

#### 2.2.1.5 Ereignisgesteuerte Systeme

Sobald Ereignisse von Unternehmen erkannt, analysiert und kommuniziert werden sowie durch Ereignisse Aktivitäten ausgelöst werden, wird von ereignisgesteuerten Unternehmen gesprochen. Die Ereignisorientierung ermöglicht den Unternehmen [21, S.28-32]:

- Entscheidungen auf Grundlage von Echtzeitdaten zu treffen,
- einfache sowie komplexe Ereignismuster direkt zu erkennen und schnell auf sie zu reagieren,
- Ereignisse unterschiedlicher Abstraktionsebenen bei Überlegungen einzubeziehen,
- große kontinuierlich einströmende Ereignisse zu verarbeiten,
- fachliche Prozesse realitätsnah abzubilden sowie
- Anwendungssysteme zu verändern und zukünftige Anforderungen mit einzubeziehen.

Die Ereignisverarbeitung erfolgt in ereignisgesteuerten Systemen. Im ersten Schritt müssen die Ereignisse unmittelbar nach ihrem Entstehen erkannt werden, um sie im zweiten Schritt verarbeiten zu können. Hierzu werden sie abstrahiert, klassifiziert oder gelöscht. Darüber hinaus erfolgt die Suche nach Beziehungen zwischen den Ereignissen. Sobald ein Muster erkannt wird, folgt eine zuvor definierte Reaktion [22, S.5ff.].

#### 2.2.1.6 Auswertungsfenster

Auf Grund der Tatsache, dass ein Ereignisstrom nicht abbricht, ist es notwendig die unendliche Menge von Ereignissen in endliche Mengen zu unterteilen. Ein gängiger Ansatz, um Ereignismuster in Teilbereichen zu finden, sind Auswertungsfenster. Dabei findet eine Unterscheidung zwischen gleitenden Längenfenstern und gleitenden Zeitfenstern statt. **Gleitende Längenfester** betrachten immer nur eine feste Anzahl von Ereignissen.

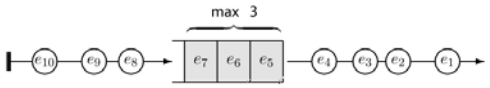



Abbildung 2.5: Gleitendes Längenfenster im Ereignistrom [33, S.35]

Trifft ein neues Ereignis in das Fenster ein, wird das älteste Ereignis entfernt. **Gleitende Zeitfenster** berücksichtigen hingegen die Ereignisse, die in einer festen Zeitdauer vorkommen [33, S.24].

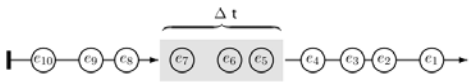



Abbildung 2.6: Gleitendes Zeitfenster im Ereignistrom [33, S.34]

Ein **Sliding Window** enthält einen Verschiebefaktor. Dieser gibt an mit welcher Anzahl von Ereignissen oder mit welchem Zeitintervall sich das Fenster bewegen soll. Je nachdem wie groß der Faktor gewählt ist, kann es zu Überlappungen, Lücken oder direkt aufeinander folgenden Fenstern kommen. Beim **Landmark Window** wird der konkrete Zeitpunkt des Fensters angegeben [33, S.33ff.].

### 2.2.2 Die Bedeutung von Echtzeit

Auf Grund der Halbwertzeit vieler Informationen, müssen Unternehmen auf Ereignisse so schnell wie möglich reagieren. Der Zeitpunkt des Wertverlustes hängt dabei stark von dem Inhalt des Datums ab. Grundsätzlich kann festgehalten werden, dass eine unmittelbare Reaktion einen höheren Wert erzielt als eine spätere. Eine Reaktion in Echtzeit treibt den Wert auf das Maximum. Im Online Banking muss ein Betrug in Millisekunden erkannt und blockiert werden, sodass betrügerische Transaktionen unterbunden werden. Fällt der Betrug erst nach der Durchführung der Transaktion auf, kann die Identifikation zwar noch eine nützliche Information für die Bank sein, aber der Wert ist deutlich geringer. Gleiches gilt für Kaufempfehlungen im Internet. In Echtzeit müssen das Kaufinteresse eines Kunden analysiert und Empfehlungen ausgesprochen werden. Hat der Kunde die Seite verlassen, sinken die Chancen ein Produkt an ihn zu verkaufen [55, S.111f.].

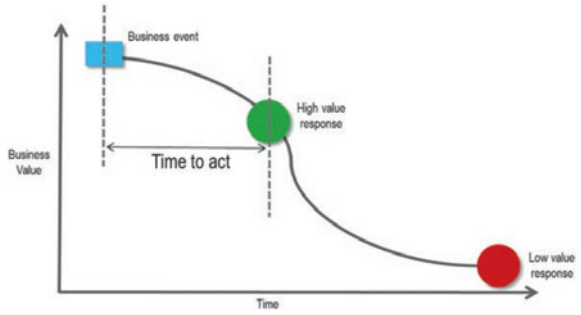



Abbildung 2.7: Wertverlust von Daten im Laufe der Zeit [55, S.112]

Echtzeitsysteme lassen sich in „harte“ und „weiche“ Systeme unterteilen. Unter „harten“ Echtzeitsystemen verstehen sich Anwendungssysteme, die nach einer gewissen Frist wirkungslos sind. „Weiche“ Echtzeitsysteme können hingegen auch nach einer gewissen Zeitspanne noch einen Nutzen haben. In der Forschung und Lehre wird meistens nur die „harte“ Echtzeit berücksichtigt.

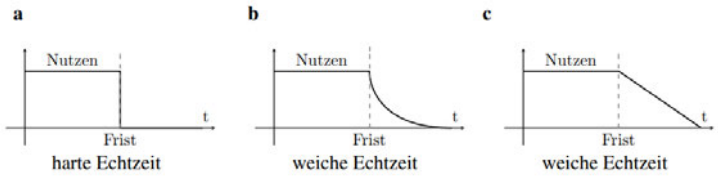



Abbildung 2.8: Harte und Weiche Echtzeit [19, S.358]

Die Deutsche Industrienorm (DIN) 44300 definiert Echtzeit als „[...] den Betrieb eines Rechensystems, bei dem Programme zur Verarbeitung anfallender Daten ständig betriebsbereit sind, derart, dass die Verarbeitungsereignisse innerhalb einer vorgegebenen Zeitspanne verfügbar sind."

### 2.2.3 Event-Driven Architecture mit Complex Event Processing

Um den Anforderungen einer dynamisch vernetzten Welt gerecht zu werden, bedarf es einer Anwendung, die konstant verfügbar ist und die auf die vielfältigen Signale der Umwelt reagieren kann [33, S.1]. Dieses bietet die nun folgend erläuterte Event-Driven Architecture.

#### 2.2.3.1 Event-Driven Architecture

Die Softwarearchitektur eines Systems gibt Auskunft über ihre Komponenten, deren Beziehungen sowie Interaktionen [22, S.15]. Die für die Ereignisorientierung und Echtzeitverarbeitung entwickelte Architektur nennt sich Event-Driven Architecture (EDA), bei der die Verarbeitung von Ereignissen im Mittelpunkt steht. Die Ereignisse werden aus den Ereignisströmen unterschiedlicher Quellen identifiziert, analysiert sowie modifiziert und Muster zwischen ihnen ermittelt. Durch den Austausch von Ereignissen kommunizieren die Komponenten miteinander [21, S.28].

Die EDA lässt sich in drei Schichten unterteilen: die Ereignisquellen, die Ereignisverarbeitung und die Ereignisbehandlung.

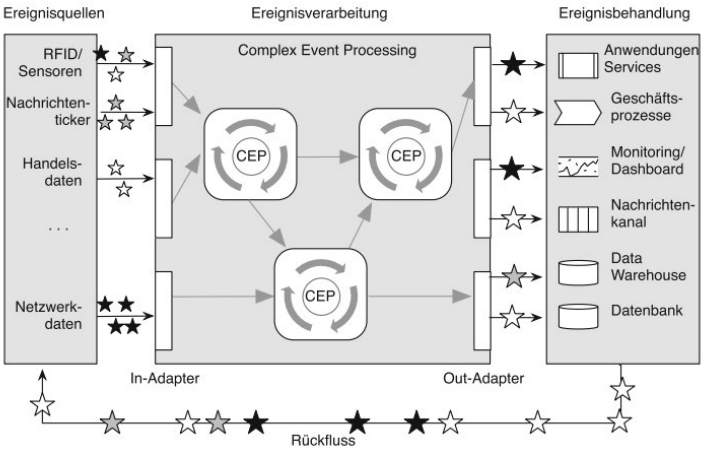



Abbildung 2.9: Schichten einer Event-Driven Architecture [22, S.15]

Zu den **Ereignisquellen** gehören Module, die Ereignisse erkennen oder Ereignisse aus relevanten Informationen generieren. Ereignisquellen senden ihre Ereignisse unmittelbar nach ihrer Identifikation oder Generierung an einen Mediator weiter. Während die Ereignisquelle weder den Ort der Weiterverarbeitung des Ereignisses kennt, noch weiß ob weitere Komponenten zur Verarbeitung existieren, hat der Mediator das entsprechende Wissen und ist für die Verteilung der Ereignisse zuständig. Beispiele für Ereignisquellen können Sensoren in Maschinen oder Benutzerinteraktionen sein.

Die **Ereignisverarbeitung** arbeitet als Ereignissenke. Sie empfängt die Ereignisse und verarbeitet sie umgehend mittels Complex Event Processing. Erkannte Muster führen zur Einleitung von definierten Reaktionen. Die eigentliche Ausführung findet im Anschluss in der **Ereignisbehandlungsschicht** statt. Beispielhafte Reaktionen auf ein Muster können das Aufrufen von Diensten, die Aktualisierung eines Dashboards oder das Veröffentlichen einer Nachricht sein. Die Ereignisbehandlung kann wieder zu neuen Ereignissen führen, die erneut verarbeitet werden müssen [22, S.15f.].

Da ausschließlich die Ereignisverarbeitung die Fachlogik kennt, reduzieren sich die Abhängigkeiten zwischen Quellen und Senke auf ein Minimum. Lediglich die syntaktische Struktur und die Semantik der Ereignisse müssen aufeinander abgestimmt werden.

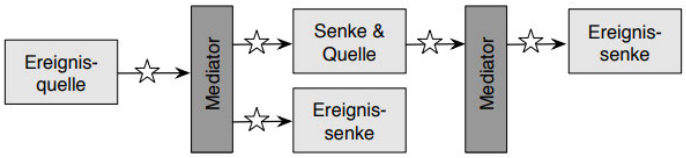



Abbildung 2.10: Verarbeitungsmodell mit Ereignisquellen, -senken und Mediatoren [21, S.52]

Es ergibt sich eine **lose Kopplung** zwischen den interagierenden Komponenten, die es der EDA ermöglicht, die Ereignisverarbeitung durch weitere Komponenten zu ergänzen oder bestehende Komponente zu verändern, ohne dass die Ereignisquellen modifiziert werden müssen.

Die **asynchrone Kommunikation** zwischen den Ereignisquellen und der Ereignissenke führt dazu, dass weder Quellen noch die Senke aufeinander warten müssen. Es ist irrelevant, ob die Ereignissenke zum Zeitpunkt des Ereignissendens der Quellen verfügbar ist. Der Mediator speichert die gesendeten Ereignisse solange bis sie dem Empfänger zugestellt werden können.

Typischerweise erfolgt der Austausch von Ereignisnachrichten über den **Publish-/ Subscribe Modus**, d.h. die gesendeten Ereignisnachrichten sind Instanzen eines spezifischen Typs, die von Empfängern abonniert werden können. Sobald der Mediator eine Nachricht von der Ereignisquelle erhält, informiert er alle auf diesen Nachrichtentypen abonnierten Empfänger, die darauf die Nachricht abrufen können.

Da die Ereignisquellen ausschlaggebend für den Nachrichtenaustausch sind, wird von einem **Push-Modus** gesprochen [21, S.51ff.].

#### 2.2.3.2 Complex Event Processing

Für die Ereignisverarbeitung der EDA werden Complex Event Processing (CEP) Komponenten eingesetzt. Das CEP ist eine Softwaretechnologie zur Verarbeitung kontinuierlicher Ereignisströme in Echtzeit. Während die Datenverarbeitung bei herkömmlichen Datenbanktechnologien statisch erfolgt, also die Daten persistent und indexiert gespeichert und im zweiten Schritt mit verschiedenen Abfragen zugänglich gemacht werden,

treffen beim CEP kontinuierlich dynamische Daten ein, die auf festgelegte Muster durchsucht werden [21, S.57f.] [33, S.3].

Die Grundlage für die Datenverarbeitung legen die definierten Regeln. Eine Ereignisregel hat zwei Aufgaben. Sie muss ein Muster zunächst erkennen und im zweiten Schritt eine definierte Reaktion einleiten. Hierfür besteht sie aus einem Bedingungsteil, der den Datenstrom nach vorliegenden Mustern durchsucht und einem Aktionsteil, der für die Reaktionseinleitung zuständig ist, sobald ein Muster erkannt wurde. Die Event Processing Language (EPL) stellt die Sprache, in der die Regel formuliert werden kann, bereit [22, S.10ff.].

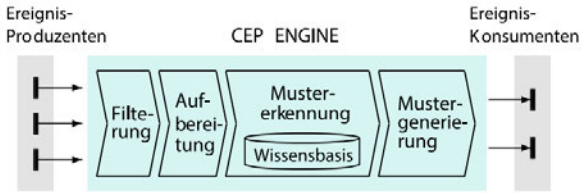



Abbildung 2.11: Prinzip des Complex Event Processing [33, S.3]

Bevor die Mustererkennung beginnt, werden irrelevante Ereignisse aussortiert. Diese können anhand ihrer Typen oder Attributwerten erkannt werden. Des Weiteren können Duplikate als überflüssige Ereignisse herausgefiltert werden. Nach der Filterung der Ereignisse folgt die Aufbereitung. Nicht relevante Bestandteile der Ereignisobjekte können eliminiert, fehlerhafte Ereignisdaten korrigiert, fehlende Informationen, die mit Hilfe von Hintergrundwissen hergeleitet werden, hinzugefügt und Attributwerte an die verwendete CEP Engine angepasst werden. Beispielsweise müssen Einheiten umgewandelt oder Formate überführt werden [33, S.27ff.].

#### 2.2.3.3 Event Processing Agents

Eine CEP-Komponente bzw. ein Event Processing Agent (EPA) besteht aus dem Ereignismodell, den Ereignisregeln und der Event Processing Engine [21, S.66ff.]. Mit dem **Ereignismodell** werden die zu verarbeitenden Ereignisse spezifiziert. Dies ist wichtig, da nur ein einheitliches Ereignisformat zwischen den Komponenten ausgetauscht werden kann. Um dennoch eine hohe Entkopplung zwischen den Quellen und der Senke zu erhalten, ist es notwendig, dass das Ereignis alle entscheidenden Informationen mit sich führt.

Die **Ereignisregeln** werden zur Verarbeitung der Ereignisse benötigt. Sie enthalten die Informationen der zu suchenden Muster. Außerdem geben sie an, wie auf die jeweilige Musteridentifikation reagiert werden soll. In die **Event Processing Engine** fließt der Ereignisstrom kontinuierlich ein. Die Engine ist für die eigentliche Ausführung zuständig. Sie verwendet die Ereignisregeln, um Muster im Strom zu erkennen und entsprechende Aktionen in Gang zu setzen. Die Ereignisdaten müssen so lange im Arbeitsspeicher gehalten werden, bis ihr Fenster abgelaufen ist. Nur so kann dafür gesorgt werden, dass auch vergangene Ereignisse bei der Mustererkennung berücksichtigt werden können.

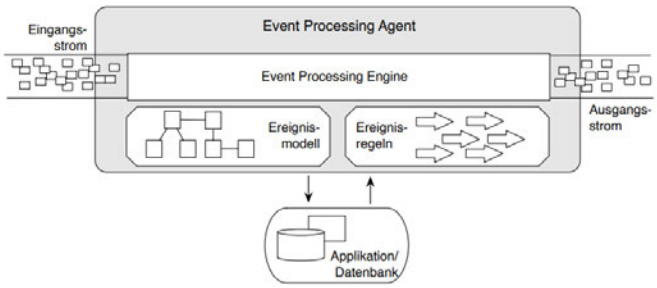



Abbildung 2.12: Elemente eines Event Processing Agent [21, S.66]

Mehrere Event Processing Agents, die sich während der Ereignisverarbeitung austauschen, bilden ein **Event Processing Network**. Event Processing Networks werden benötigt, um komplexe Problemstellungen handhabbar zu machen. „Eine CEP-Anwendung besteht [...] aus einem Netzwerk von interagierenden, leichtgewichtigen Event Processing Agents bzw. CEP-Komponenten.“ [21, S.69]

Jeder einzelne Event Processing Agent bedient dabei einen abgegrenzten Teilschritt der gesamten Ereignisverarbeitung und wird aufgrund der eingeschränkten Aufgabe als leichtgewichtig bezeichnet. Eine Ausgabe einer Komponente kann dabei als Eingabe einer anderen dienen.

#### 2.2.3.4 Fähigkeiten einer Event-Driven Architecture mit CEP

Zu den Fähigkeiten einer Event-Driven Architecture mit CEP gehören die Echtzeitfähigkeit, das Erkennen von Ereignismustern, die Verarbeitung von Ereignisströmen sowie das in Regeln festgehaltene Wissen. CEP Systeme sind in der Lage die großen kontinuierlich eintreffenden Ereignismengen verschiedener Quellen in Echtzeit auszuwerten und zu

verarbeiten. Dabei können sie den Ereignisstrom mit unterschiedlichen Mustern abgleichen. Die Komplexität reicht dabei von einfachen Ereignismustern bis hin zu Mustern bestehend aus komplexen Ereignissen auf höheren Abstraktionsebenen. Um komplexere Muster zu erkennen, müssen die Ereignisse häufig um Kontextwissen ergänzt werden. WENN-DANN-Regeln enthalten das Wissen wie die Ereignisse verarbeitet werden sollen. Tritt das spezifizierte Ereignismuster im Bedingungsteil ein, folgt die im Aktionsteil beschriebene Handlungsmaßnahme (WENN Ereignismuster DANN Aktion) [21, S.58ff.].

#### 2.2.3.5 Bewertung der Event-Driven Architecture

Die Grundlage für die Bewertung von EDA bilden ihre wichtigsten Eigenschaften: Ereignisaustausch als Kommunikationsmittel, Publish-Subscribe-Interaktion mit Push-Modus sowie Asynchronität als Kommunikationsmuster, Event Processing Network mit leichtgewichtigen Komponenten zur Verarbeitung von komplexen Ereignissen und definierte Ereignismuster und -regeln als Wissensrepräsentation.

Qualitätsmerkmale einer EDA:

- **Aktualität**: Da die Ereignisse umgehend nach ihrem Eintreten verarbeitet werden, liefert die EDA ein realitätsnahes Abbild.
- **Agilität**: Die lose Kopplung zwischen den Komponenten unterstützt flexible Anpassungen. Ereignisquellen und -typen sowie CEP-Komponente können hinzugefügt, entfernt oder modifiziert werden.
- **Effizienz**: Sowohl Event Processing Engines, die mehrere hunderttausend Ereignisse pro Sekunden verarbeiten können, als auch ein gut gestaltetes Event Processing Network tragen dazu bei große Mengen von Ereignissen effizient zu verarbeiten.
- **Robustheit**: Die geringen Abhängigkeiten zwischen den Komponenten sorgen neben der agilen Anpassungsmöglichkeit dafür, dass Änderungen nur an eindeutig definierten Stellen notwendig sind.
- **Wartbarkeit**: Durch die interagierenden leichtgewichtigen CEP-Komponenten, das in Regeln festgehaltene Wissen und die lose Kopplung der Systemteile wird die Wartbarkeit der EDA erhöht.
- **Skalierbarkeit**: Durch die in der EDA verwendeten Kommunikationsmuster werden die Systemkomponenten entkoppelt und ermöglichen somit eine physikalische Verteilung.

Potenzielle Defizite eines ereignisgesteuerten Ansatzes:

- **Redundanz**: Aufgrund der losen Kopplung, müssen dieselben Daten und Funktionalitäten in mehreren Systemen vorhanden sein. Das hat zur Folge, dass Änderungen an den Daten an mehreren Stellen vollzogen werden müssen und somit die Wartbarkeit erschwert wird.
- **Schwierige Fehlersuche**: Ein strukturiertes Testen von EDA Anwendungen erweist sich durch die lose Kopplung als schwierig. Sowohl die Lokalisierung als auch die Behebung von Fehlern werden erschwert.
- **Undurchsichtiger Kontrollfluss**: Durch zu viele Regelmengen kann die Übersichtlichkeit und damit die Kontrolle des Systemverhaltens verloren gehen.

Auf Grund der Tatsache, dass die Anwendungsprobleme in Unternehmen sehr komplex und vielfältig sind, bietet sich meistens eine Kombination aus konventionellen und ereignisgesteuerten Softwarearchitekturen an. Die Event-Driven Architecture kann dabei insbesondere dazu beitragen die Agilität, die Reaktionsfähigkeit und die Echtzeitfähigkeit zu erhöhen [21, S.72-76].

### 2.2.4 Anwendungsbereiche

Während konventionelle Architekturen strukturierte Geschäftsprozesse ohne Echtzeitanforderungen gut unterstützen, sind ereignisorientierte Softwarearchitekturen für die Erfüllung folgender Anforderungen geeignet:

- **Komplexe Fachlogik**: Sie unterstützen komplexe Verarbeitungsprozesse wie z.B. das Auffinden von Zusammenhängen aktueller und historischer Daten, das Erkennen von Ereignismustern oder die Abstraktion von Daten auf verschiedenen Ebenen.
- **Große Datenvolumina**: Sie sind in der Lage große Mengen von kontinuierlich einfließenden Daten zu verarbeiten.
- **Geringe Latenzzeit**: Sie reagieren auf eintretende Ereignisse in Echtzeit.
- **Skalierbarkeit**: Sie ermöglichen eine physikalische Verteilung, die durch das steigende Daten- und Transaktionsaufkommen notwendig ist.
- **Agilität**: Sie begünstigen einfache Änderungen, die sich aus den schnell verändernden fachlichen Entscheidungen ergeben.

Das Monitoring, verschiedene Sensornetze sowie Analysen sind Beispiele für Anwendungsbereiche, die von diesen Eigenschaften geprägt sind. Das Monitoring soll den aktuellen Zustand von Prozessen bzw. Gesamtsystemen aufzeigen, um diese überwachen zu können. Selten reichen hierfür einzelne Daten, stattdessen müssen in Echtzeit eine Vielzahl von Ereignissen ausgewertet werden. In Sensornetzwerken werden permanent große Datenmengen von Ereignissen erzeugt. Die Sensoren sind dabei für die kontinuierliche Weiterleitung der Ereignisse der physikalischen Welt an die Unternehmensanwendungen zuständig. Um rechtzeitig auf Entwicklungen reagieren zu können, müssen atomare Ereignisse kontinuierlich und im direkten Anschluss ihres Auftretens auf Zusammenhänge und Muster analysiert werden [21, S.39-44].

Anwendung finden Echtzeitsysteme in den Bereichen Smart Home, der Überwachung von Fahrzeugen, der Medizin und im Bereich der Geschäftsprozesse, welcher für diese Thesis von Interesse ist.

Um Geschäftsprozesse beobachten und bewerten zu können, werden aussagekräftige Kennzahlen verwendet. Diese werden kontinuierlich gemessen und geben Aufschluss, ob Prozesse in bestimmten Situationen fortgesetzt werden sollen oder eine Anpassung notwendig ist. Beispielsweise sollte eine vom Kunden stornierte Bestellung nicht mehr geliefert werden oder eine durch den Transport verspätetet Lieferung zu keinen Engpässen in der Produktion führen.

Neben den aufgeführten Anwendungsfeldern gibt es noch zahlreich viele weitere Beispiele bei denen Echtzeitsysteme notwendig sind. Zu den klassischen zählen auch der Wertpapierhandel, die Betrugserkennung im Onlinebanking sowie Wartungszeitpunkte von Maschinen [33, S.5-13].

Die unterschiedlichsten Anwendungsfelder weisen darauf hin, dass Echtzeitsysteme bereits heute einen großen Mehrwert bieten und in Zukunft vermutlich eine höhere Relevanz in Unternehmensarchitekturen haben werden.

# 3 Analyse

Das Kapitel Analyse dient zur Vorbereitung der Konzeption des zu entwickelnden Prototyps. Behandelt werden u.a. die zu berücksichtigenden Daten und personalisierbare Maßnahmen für eine E-Commerce Plattform. Zum Abschluss werden alle relevanten Anforderungen an den Prototypen gebündelt dargestellt.

Bei einem Blick zurück auf die Geschichte des E-Commerce (siehe Kapitel 2.1.1.2) zeigt sich, dass die Datensammlung, -analyse sowie -nutzung seit einigen Jahren eine maßgebliche und wachsende Rolle beim Verkauf von Waren über das Internet spielt. Das Fundament der Datenverarbeitung bildet hierbei Big Data (siehe Kapitel 2.2.1.1), das u.a. durch die große Menge an Traffic Daten im Internet entsteht. Auf diese Daten sollte ein Unternehmen so schnell wie möglich reagieren können, um den Wert der Daten nicht zu verlieren (siehe Kapitel 2.2.2). Denn hat ein Kunde eine Website ersteinmal verlassen, sinken die Chancen ihm ein Produkt zu verkaufen. Um sich gegenüber der Konkurrenz im Internethandel durchzusetzen, ist neben der schnellen Reaktion auf eintreffende Ereignisse vor allem wichtig, die Customer Experience zu maximieren. Durch eine Unterstützung bei Suchabläufen lässt sich eine Informationsüberlastung beim Kunden mindern (siehe Kapitel 2.1.2.3). Kaufentscheidungen werden erleichtert und dadurch höhere Verkaufszahlen erzielt.

Grundsätzlich lässt sich festhalten, dass für die Entwicklung eines E-Commerce Systems eine Analyseform benötigt wird, die große Datenmengen personalisiert und schnellstmöglich verarbeiten kann.

## 3.1 Analyse von Daten

Im Zentrum jeder Analyse stehen die zu verarbeitenden Daten, die in diesem Kapitel herausgearbeitet werden sollen. Im Folgenden werden der Aufbau und Inhalt von Strea-

ming Daten, Kunden- und Produktdaten sowie die für eine Segmentierung notwendigen Daten näher untersucht.

### 3.1.1 Streaming Daten

Zunächst wird der Fokus auf die kontinuierlich einfließenden Datenströme gelegt, da diese bei einer Echtzeitanalyse von größter Bedeutung sind. Als zentraler Input zählen die Kundeninteraktionen, die durchweg auf der Website entstehen. Das Analysieren des Kundenverhaltens soll als Ziel haben, die Intentionen des Kunden zu verstehen, Interessen und zukünftige Bedürfnisse daraus abzuleiten und personalisierte Produktempfehlungen und Angebote erstellen zu können. Des Weiteren können ergänzende Datenquellen in das System einfließen, die ebenfalls in die Empfehlungen und Angebotserstellung einfließen können.

Zur Herleitung der Struktur der Traffic Daten, hilft es sich vorerst Gedanken über ihren Inhalt zu machen. Hierfür ist es sinnvoll zwischen verschiedenen Klick-Typen zu unterscheiden. Ein Kunde kann in einem online Shop mit Hilfe eines Suchfelds, einer Navigationsleiste oder mittels Werbebanner sowie über Produktvorschläge die für ihn interessanten Produkte erreichen. Auf der Suchergebnis- bzw. Kategorieseite kann er Produkte filtern und unterschiedlich sortieren lassen. Hat ein Produkt sein Interesse geweckt, gelangt er durch einen Klick auf die Produktdetailseite. Hier besteht die Möglichkeit, verschiedene Varianten zu betrachten und das Produkt zu einer Merkliste oder dem Warenkorb hinzuzufügen. Oft besteht außerdem die Möglichkeit, Empfehlungen für ein Produkt zu hinterlassen. Favorisierte Produkte lassen sich in einer gespeicherten Liste anzeigen und Produkte im Warenkorb durch weitere Klicks sowie der Hinterlegung von Zahlungsinformationen erwerben [18, S.68-75]. Diese nicht vollständige Beschreibung von Kundeninteraktionen im online Shop verdeutlich die Menge und Vielfalt an Klick-Typen bzw. Ereignissen.

Genau wie das IT- und Beratungsunternehmen IBM [8] in einem Beispiel zeigt, wird sich die folgende Ausarbeitung, auf die in der Tabelle 3.1 aufgelisteten Typen konzentrieren.

| | |
|---|---|
| **Typ Anschauen:** | Page View (Produkt anschauen) |
| **Typ Merken:** | Add To Cart (Produkt dem Warenkorb hinzufügen) |
| **Typ Kaufen:** | Order Event (Produkt kaufen) |

Tabelle 3.1: Verwendete Klick-Typen

Wie aus dem Kapitel Ereignis (siehe Kapitel 2.2.1.3) hervorgeht besteht ein Ereignis neben dem **Typ** mindestens auch aus einem **Zeitstempel** und einer **Ereignis ID**.

Betrachtet man zunächst nur die Ereignisse rund um die Kundeninteraktion erweisen sich zusätzlich die IP-Adresse, eine Session-ID, die Page URL sowie wenn vorhanden die Kunden-ID bzw. User-ID als sinnvoll. Darüber hinaus können Informationen wie die Ausgangs-URL sowie technische Details, wie der verwendete Browser, das Ereignis weiter beschreiben [17, S.26f.].

- **IP-Adresse:** Mit Hilfe der IP-Adresse können auch nicht eingeloggte Kunden identifiziert und passend angesprochen werden.
- **Session-ID:** Die Session-ID verbindet die einzelnen Ereignisse des Kunden und hilft somit das Verhalten über einen gesamten Besuch zu verstehen.
- **Page-URL:** Mit Hilfe der Page-URL wird der Verlauf der betrachteten Seiten im Shop erkennbar. Sie enthält außerdem den Namen der Seite, der Aufschluss zu angeschauten Kategorien oder Produkten geben kann.
- **Kunden-ID bzw. User-ID:** Um den Kunden besser verstehen zu können, sind historische Daten relevant, die über die Kunden-ID zugänglich gemacht werden können.

Werden weitere Quellen angebunden, ist es zudem hilfreich, die eintreffenden Ereignisse mit einem Quellenverweis zu ergänzen. In dem zu entwickelnden Prototypen können zusätzliche Quellen die Nachfrage nach Produkten, die sich ständig ändernden Lagerbestände oder konkurrierende Angebote sein. Bei den aus dem Unternehmen zur Verfügung gestellten Daten wie die Lagerbestände können Ereignistypen und zusätzliche Informationen wie die Produkt-ID im Vorhinein konstruiert werden. Externe Ereignisse wie konkurrierende Angebote müssen hingegen zunächst mit einer entsprechenden Programmierschnittstelle (API) verfügbar gemacht und anschließend angereichert werden. Ihre Struktur ist stark abhängig von den zur Verfügung gestellten Daten.

### 3.1.2 Kunden und Produktdaten

Genau wie die Kundeninteraktionen tragen auch vorhandene Kunden- und Produkt-Stammdaten dazu bei, die Kunden zu verstehen, Wünsche und Eigenschaften zu identifizieren und somit Empfehlungen und Angebote personalisiert stellen zu können.

#### 3.1.2.1 Kundendaten

Bei einer typischen Webshop Registrierung müssen der Name, eine E-Mail-Adresse, ein Passwort sowie häufig das Geschlecht und Geburtsdatum angegeben werden. Bei einer Bestellung sind zusätzliche Angaben wie die Lieferadresse und Zahlungsinformationen notwendig [23, S.532].

Der Fashionshop About You [1] ermöglicht zusätzlich, freiwillige Angaben zu Lieblingsmarken, Größen sowie Einstellungen zur Zustimmung zu Benachrichtigungen zu hinterlegen. Unter dem Menüpunkt "Deine Bestellungen" werden vergangene Bestellungen mit Informationen zu dem Bestelldatum, den gelieferten und retournierten Produkten sowie der Bezahlart, der Lieferadresse und Rechnungsadresse gespeichert. Favorisierte Produkte sind auf der Wunschliste gespeichert und auch hinterlegte Produkte im Warenkorb bleiben zugänglich. Dieser kleine Einblick verbildlicht einige Informationen, die zu registrierten Kunden gespeichert werden.

Experten empfehlen, die Daten nicht in einer einzelnen Tabelle zu speichern. Schwering [50] zeigt in seinem Vortrag „Using Flink with MongoDB to enhance relevancy in personalisation" auf der Flink Forward Messe 2015 eine beispielhafte Unterteilung der zu speichernden Kundendaten:

- Ein **Kundenprofil** könnte demnach aus allgemeine Kundendaten wie Name, Adresse, Geschlecht, Geburtsdatum bestehen.
- In den **Kunden Daten** könnten die Bestell- und Werbe-Historie enthalten sein.
- **Session Daten** könnten die betrachteten, favorisierten sowie im Warenkorb hinterlegten Produkte enthalten.
- Zu den **Personalisierten Daten** könnten präferierte Marken, Farben oder Größen sowie personalisierte Empfehlungen gespeichert sein.

#### 3.1.2.2 Produktdaten

Ohne gut gepflegte Produktdaten können weder das Interesse des Kunden nachvollzogen noch geeignete Produktempfehlungen vorgeschlagen werden. Um eine Vorstellung zu gespeicherten Produktinformationen zu erhalten ist es hilfreich einen Blick auf einen Webshop zu werfen.

Auf einer Produktdetailseite des Onlinehändlers OTTO [46] lässt sich bereits an Hand der URL der Name sowie die Produkt- und Varianten-ID erkennen. Zusätzlich veranschaulichen Fotos oder vereinzelt auch Videos das Produkt. Über dem Foto wird der Pfad des Produktes im Shop angezeigt. Aus ihm lassen sich zugeordnete Kategorien schließen. Darunter befindet sich die Marke und der Name des Produktes. Außerdem kann zwischen verschiedenen Farben und Größen ausgewählt werden. Der Preis sowie eventuelle Reduzierungen und Kundenbewertungen gehören ebenfalls zu den zentralen Produktinformationen. Weitere Detail- und Artikelbeschreibungen befinden sich weiter unten auf der Seite.

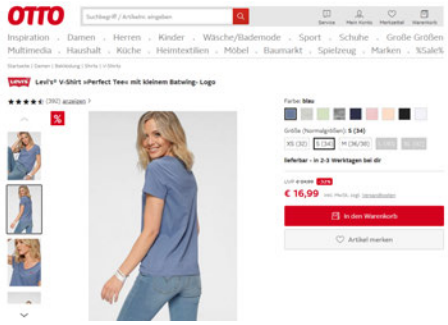



Abbildung 3.1: Produktdetailseite auf Otto.de [45]

Wie bei den Kundendaten kann auch hier festgehalten werden, dass es nicht immer sinnvoll ist, alle Informationen in einer Produkt Tabelle abzubilden. Beispielsweise ist es üblich, nur die IDs zu Kategorien und Marken zu speichern, die dann ihre eigene Tabelle mit näherer Beschreibung besitzen. Auch Produktbewertungen werden in einer eigenen Tabelle gespeichert.

### 3.1.3 Clustering

Aus der in den Grundlagen bereits vorgestellten Kundenclusterung (siehe Kapitel 2.1.5) wird deutlich, dass eine Gruppierung von Kunden ebenfalls dabei helfen kann, personalisierte Angebote und Empfehlungen zu erstellen. Eigenschaften und Interessen der Kundengruppen werden aus dem Kundenverhalten ermittelt und dementsprechend Bedürfnisse und Wünsche abgeleitet.

Der Echtzeitverarbeitung nachgelagert kann das aktuelle Kundenverhalten in Verbindung mit vorhandenen Stammdaten ausgewertet werden. Hierbei stellt sich die Frage nach den

Kriterien zur Bildung der Cluster. Im Idealfall werden alle verfügbaren Informationen über Kunden in die Analyse mit einbezogen und somit sowohl die verhaltensorientierten Merkmale, die soziodemografischen Merkmale, als auch die psychografischen Merkmale mit einbezogen.

Während **verhaltensorientierte Kriterien**, wie die Sortiment- und Marken-Präferenz oder die Preisklassen in der der Kunde einkauft, und die **soziodemografischen Kriterien**, wie das Alter, Geschlecht, der Wohnort des Kunden, aus den oben benannten Datenquellen zur Verfügung stehen sollten, müssen die **psychografischen Kriterien**, wie die Kaufabsicht oder das Shoppingerlebnis, zunächst aus bereits vorhandenen Daten ermittelt werden.

Um bei der Analyse mit einbezogen werden zu können, müssen die Clusterergebnisse dem System schließlich zugängig gemacht werden.

## 3.2 Personalisierte Angebotsentwicklung

Das Kapitel Angebotsentwicklung soll die zentralen Ideen der Personalisierung im Online Shop auf Basis der in Kapitel 3.1 genannten Daten vorstellen. Die Personalisierung wird in Angebote, Produktempfehlungen sowie in Seiten-Layouts unterteilt.

### 3.2.1 Angebote

Im Folgenden werden beispielhafte Angebote inklusiver ihrer Anforderungen für verschiedene Kundengruppen aufgelistet:

- Kunden, die bisher noch keinen Newsletter abonniert haben, erhalten den Hinweis, sich mit einem entsprechenden Abonnement einen Rabatt Code sichern zu können.
- Je Kundengruppe gibt es spezifizierte Angebotstage mit unterschiedlichen Angeboten. Während Schnäppchenjägern Rabatte auf bereits reduzierte Artikel gewährt werden, erhalten Kunden, die an den neusten Trends interessiert sind, auf aktuellere Artikel eine Reduzierung.
- Kundengruppen wie Neukunden oder Studenten erhalten einen extra Rabatt.
- Kunden, die bereits einige Produkte im Warenkorb liegen haben, erhalten das Angebot, ab einer gewissen Anzahl gekaufter Produkte ihren gesamten Einkaufspreis um 15% gesenkt zu bekommen.

- Kunden, die häufiger im Jahr bestellen, wird ein kostenloser Versand geboten.
- Interessieren sich vermehrt Kunden für ein bestimmtes Produkt, werden sie gegebenenfalls auf die geringe Verfügbarkeit hingewiesen.

Mit Hilfe der verfügbaren Daten können die einzelnen Kunden auf die Anforderungen der vorhandenen Angebote geprüft und im Idealfall mit passenden Hinweisen vom Kauf überzeugt werden. Während der Kunde den Shop durchstöbert, werden Wahrscheinlichkeiten zum Kaufabschluss aus der Reihenfolge der Kundeninteraktionen ermittelt und entsprechende Angebote erstellt. Damit die Angebote nicht zur Selbstverständlichkeit werden, dürfen Kunden jedoch nicht mit Angeboten überschüttet werden. Eine gezielte Angebotsauswahl muss von dem System schnell getroffen und bei späteren Analysen mit einbezogen werden.

### 3.2.2 Empfehlungssysteme

Um der Informationsüberflutung des Kunden entgegenzuwirken und den Kaufprozess zu erleichtern, werden Produktempfehlungssysteme eingesetzt. Der online Versandhändler Amazon [6] arbeitet mit verschiedenen Typen von Empfehlungssystemen. Auf der Produktseite eines Smartphones [5] werden bspw. Empfehlungssysteme auf Basis von fünf Kriterien eingebunden.

Zunächst wird die Cross Selling Methode eingesetzt. Unter dem Titel „Wird oft zusammen gekauft“ wird zum Produkt passendes Zubehör angeboten. Darunter folgen zwei gesponserte Empfehlungsbanner mit den Namen „Verwandte Produkte zu diesem Artikel“ sowie „4 Sterne und mehr“, bevor daraufhin ein Vergleich zu fünf ähnlichen Produkten folgt. Als letztes Empfehlungssystem schließen sich erneut Vorschläge für ähnliche Produkte an, allerdings nun auf der Basis von anderem Kundenverhalten: „Kunden, die diesen Artikel angesehen haben, haben auch angesehen“.


Abbildung 3.2: Amazon Emfehlungssysteme [5]

Wie aus diesem Beispiel und den in den Grundlagen vorgestellten Empfehlungssystemen (siehe Kapitel 2.1.4) hervorgeht, basieren Empfehlungen auf unterschiedlichen Kriterien. Zum einem können Zusatzprodukte empfohlen werden, die sich aus der Kaufhistorie von häufig zusammengekauften Artikeln ergeben, zum anderen können Produktalternativen vorgeschlagen werden. Diese resultieren aus einem ähnlichen Suchverhalten anderer Kunden oder aus ähnlichen Produkteigenschaften. Empfehlungssysteme treten nicht nur auf der Produktdetailseite auf, sondern helfen auch von der Startseite aus, ein gesuchtes Produkt zu finden. Produktvorschläge können abhängig von den unterschiedlichsten Merkmalen sein. Sie können bspw. von zuletzt angeschauten Produkten sowie deren Kategorien, von der Preisklasse in der der Kunde für gewöhnlich einkauft, von Marken oder aktuellen Trends bestimmt werden. Zusätzlich können u.a. verfügbare Größen und Lagerbestände die Empfehlungen beeinflussen.

### 3.2.3 Seiten-Layout

Neben den personalisierten Aktionen und Produktempfehlungen kann auch der Aufbau einer Website personalisiert werden (siehe Kapitel 2.1.2.2). Zusätzlich zu einer namentlichen Ansprache kann die Seite für die jeweilige Kundengruppe modifiziert werden.

Orientiert man sich bspw. an den in den Grundlagen nach Bernhard und Mühling [18] eingeteilten Kundentypen Jäger, Sucher und Sammler (siehe Kapitel 2.1.5.4), könnten

- **Jäger-Typen**, die das Ziel verfolgen, möglichst schnell und mit geringem Aufwand ihr gesuchtes Produkt zu kaufen,
- **Sucher-Typen**, die den Webshop besuchen, um sich über ein bestimmtes Produkt zu informieren und dieses mit ähnlichen Produkten zu vergleichen, oder
- **Sammler-Typen**, die sich von der Vielfalt im Shop inspirieren lassen wollen,

z.B. eine unterschiedliche Anzahl und verschiedene Arten von Produktempfehlungssystemen auf den einzelnen Seiten geboten werden.

Auch ist ein differenzierter Aufbau je Device, Geschlecht, Alter oder typisch gekaufter Preisklasse denkbar. Einer älteren Person würde ein möglichst einfacher Aufbau mit großen sehr intuitiven Klick-Möglichkeiten und eine Unterstützung durch Chat-Bots zur Verfügung stehen und einer Person, die ausschließlich Premium Einkäufe tätigt, eine optisch hochwertige Seite angezeigt werden.

## 3.3 Anforderungen an eine Echtzeitanalyse im Rahmen einer E-Commerce Plattform

Nachdem die Analysen der benötigten Daten und zur Verfügung stehenden personalisierbaren Maßnahmen abgeschlossen sind, gilt es, konkrete Anforderungen an den Prototypen zu definieren. User Stories (US) sollen im Folgenden dabei helfen, funktionale und nicht funktionale Anforderungen zu verdeutlichen.

| | |
|---|---|
| **US1:** | Als Kunde erwarte ich 24/7 einen vollfunktionalen Webshop. |
| **US2:** | Als Kunde erwarte ich nach einem Klick innerhalb von maximal drei Sekunden eine vollständig geladene Website [51]. |
| **US3:** | Als Kunde erwarte ich für einen unbegrenzten Zeitraum Produkte in einem Webshop anschauen zu können. |
| **US4:** | Als Kunde erwarte ich stets einen modernen Webshop. |
| **US5:** | Als Kunde erwarte ich, dass mir nach fünf aufgerufenen Produkten mindestens eins gefällt. |
| **US6:** | Als Kunde erwarte ich Produkte vorgeschlagen zu bekommen, die mich mindestens zu 80% interessieren. |
| **US7:** | Als Kunde möchte ich über aktuelle Angebote und Rabattaktionen informiert werden. |
| **US8:** | Als Kunde möchte ich den besten Preis für ein Produkt bezahlen. |
| **US9:** | Als Kunde erwarte ich bei einem angezeigten Produkt, dass es auch verfügbar ist. |
| **US10:** | Als Kunde erwarte ich, dass sich der Webshop an relevante von mir preisgegebene Informationen wie meine Adresse oder Rechnungsinformationen beim nächsten Einkauf erinnert. |
| **US11:** | Als Kunde erwarte ich, dass sich der Webshop meine favorisierten und im Warenkorb befindlichen Produkte sowie meine abgeschlossenen Käufe merkt. |

Tabelle 3.2: User Stories aus der Rolle eines Kunden

### 3.3.1 Funktionale Anforderungen

Abgeleitet aus den User Stories ergeben sich konkrete funktionale Anforderungen an den Prototypen. Um **US5** gerecht zu werden, ist es notwendig, das Kundenverhalten in Echtzeit analysieren und nachvollziehen zu können. Aktuelle Kundeninteraktionen

müssen auf Verhaltensmuster untersucht und daraufhin die Aufmerksamkeit des Kunden umgehend auf relevante Angebote und Produkte gelenkt werden. Als Voraussetzung gilt es, die Kunden unterscheiden und ihre Interaktionen getrennt voneinander nach Mustern untersuchen zu können.

- **FA1:** Kundeninteraktionen müssen in Echtzeit nach Verhaltensmustern untersucht werden.
- **FA2:** Kunden müssen identifiziert und ihre Interaktionen unabhängig voneinander verarbeitet werden.

Eine vollständig gepflegte und aktuelle Datenbasis bildet die Grundlage für das Gros der aufgeführten User Stories. **US6** verlangt von dem Webshop, seine Kunden und deren Wünsche sowie Interessen zu kennen und entsprechend gezielte Produktempfehlungen ausgeben zu können. Hierfür werden einerseits historisierte Kunden- und Produktdaten sowie das aktuelle Kundenverhalten analysiert. Produktdaten sollen dabei so detailliert und standardisiert beschrieben sein, dass Produktempfehlungen auf Basis gleicher Eigenschaften entwickelt werden können. Auch **US10** und **US11** können nur dann erfüllt werden, wenn auf detaillierte und vollständige Kundendaten zurückgegriffen werden kann.

- **FA3:** Detaillierte und aktuelle Kundendaten stehen dem System zur Verfügung.
- **FA4:** Standardisierte und aktuelle Produktdaten stehen dem System zur Verfügung.
- **FA5:** Auf Basis der Produktdaten lassen sich Empfehlungen ableiten.

Wie aus **US7** hervorgeht, erwarten Kunden - neben ansprechenden Produktempfehlungen - stets über neue Angebotsaktionen informiert zu werden. Als Konsequenz muss es möglich sein, die Angebotslogik einfach zu modifizieren und zu erweitern. Angebote werden nur dann beansprucht, wenn sie für den Kunden einen Mehrwert bieten. Aus diesem Grund ist es entscheidend, je nach Kundengruppe unterschiedliche Angebote zur Verfügung zu stellen. Des Weiteren muss sichergestellt werden, dass Kunden nur eine begrenzte Anzahl von Angeboten unterbreitet wird. Sonst besteht die Gefahr, dass diese für Kunden zum einen zur Selbstverständlichkeit werden und sich die Kunden zum anderen von zu vielen Benachrichtigungen belästigt fühlen.

- **FA6:** Die Angebots- und Empfehlungslogik lässt sich modifizieren und erweitern.
- **FA7:** Kunden können anhand ihrer Eigenschaften und ihres Verhaltens segmentiert werden und entsprechend passende Angebote erhalten.
- **FA8:** Ein Kunde erhält eine begrenzte Anzahl an Angeboten pro Session.
- **FA9:** Empfehlungen und Angebote werden Kunden bereitgestellt.

Um **US8** erfüllen zu können, muss eine entsprechende Programmierschnittstelle (API) mit Informationen zu Produktpreisen der Konkurrenz an den Prototypen angebunden werden. So können Angebote der Konkurrenz im Webshop unterboten und dem Kunden ein attraktiver Preis geboten werden.

- **FA10:** Ein Datenstrom externer Datenquellen kann integriert und verarbeitetet werden.

Genau wie Datenströme von externen Datenquellen verarbeitet werden können, müssen zusätzlich zu den historisierten Stammdaten und den kontinuierlich einfließenden Kundeninteraktionen interne Datenströme - wie sich ändernde Lagerbestände - beachtet werden, um **US9** gewährleisten zu können.

- **FA11:** Ein Datenstrom interner Datenquellen kann integriert und verarbeitetet werden.

Die folgende Tabelle 3.3 bündelt zusammenfassend alle funktionalen Anforderungen:

| | |
|---|---|
| | **Datenverarbeitung in Echtzeit** |
| **FA1:** | Kundeninteraktionen müssen in Echtzeit nach Verhaltensmustern untersucht werden. |
| **FA2:** | Kunden müssen identifiziert und ihre Interaktionen unabhängig voneinander verarbeitet werden. |
| | **Datenbankanbindungen** |
| **FA3:** | Detaillierte und aktuelle Kundendaten stehen dem System zur Verfügung. |
| **FA4:** | Standardisierte und aktuelle Produktdaten stehen dem System zur Verfügung. |
| | **Angebotsentwicklung** |
| **FA5:** | Auf Basis der Produktdaten lassen sich Empfehlungen ableiten. |
| **FA6:** | Die Angebots- und Empfehlungslogik lässt sich modifizieren und erweitern. |
| **FA7:** | Kunden können anhand ihrer Eigenschaften und ihres Verhaltens segmentiert werden und entsprechend passende Angebote erhalten. |
| **FA8:** | Ein Kunde erhält eine begrenzte Anzahl an Angeboten pro Session. |
| | **Schnittstellen zu nachgelagerten Anwendungen** |
| **FA9:** | Empfehlungen und Angebote werden Kunden bereitgestellt. |
| | **Anbindung weiterer Datenströme** |
| **FA10:** | Ein Datenstrom externer Datenquellen kann integriert und verarbeitet werden. |
| **FA11:** | Ein Datenstrom interner Datenquellen kann integriert und verarbeitet werden. |

Tabelle 3.3: Funktionale Anforderungen

### 3.3.2 Nicht-Funktionale Anforderungen

Des Weiteren spiegeln die User Stories einige der Qualitätskriterien nach ISO/IEC25010 wider, die als Nicht-Funktionale Anforderungen festgehalten werden [16, S.109ff.].

So lässt sich aus der **US1** ableiten, dass der Webshop hochverfügbar sein muss. Dafür ist es notwendig, eine hohe Fehlertoleranz sicherzustellen und alle Single Points of Failure zu eliminieren. Sollte es widererwartend zu einem Ausfall kommen, muss es einen Wiederherstellungsmechanismus geben, um die **Zuverlässigkeit** zu gewähren.

- **NFA1:** Der Server, auf dem der Webshop gehostet wird, garantiert eine Verfügbarkeit von mindestens 99,99%.
- **NFA2:** Das System verfügt über keine Single Points of Failure.
- **NFA3:** Es existiert ein Wiederherstellungsmechanismus, der nach einem Ausfall das System erneut aufbauen kann.

**US2** und **US3** deuten auf die benötigte **Effizienz** der Anwendung hin. Hierbei spielen das Zeit- sowie das Verbrauchsverhalten eine Rolle. Sowohl die zeitlich begrenzte Wartezeit nach einer Nutzerinteraktion als auch die hohe zu verarbeitende Durchsatzmenge bilden zentrale Anforderungen an den Prototypen. Begrenzt zur Verfügung stehende Ressourcen müssen effizient ausgeschöpft werden.

- **NFA4:** Innerhalb von maximal drei Sekunden können passende Empfehlungen und gezielte Angebote analysiert und dem Kunden bereitgestellt werden.
- **NFA5:** Zur gleichen Zeit können Daten von 1000 Kunden, die im Durchschnitt drei Klicks pro Minute tätigen, verarbeitet werden.
- **NFA6:** Ein kontinuierlich eintreffender Datenstrom kann mit den vorhandenen Ressourcen verarbeitet werden.

Das Qualitätskriterium **Wartbarkeit** umfasst u.a., dass Erweiterungen ergänzt werden können und die bestehenden Funktionalitäten einfach veränderbar sind. Zusätzlich finden hierbei die Test- und Analysierbarkeit Berücksichtigung. So verlangt **US4** einen stets an die Umwelt anpassbaren Webshop, d.h. die einzelnen Komponenten des Systems müssen erweitert und verändert werden können. Die Funktionalitäten des Systems müssen testbar sein, um die Korrektheit sicherstellen und die Auslastung analysieren zu können.

- **NFA7:** Die einzelnen Komponenten der Anwendung sind unabhängig voneinander modifizier- und erweiterbar.
- **NFA8:** Das System lässt sich bezüglich seiner Funktionalitäten testen.

- **NFA9:** Das System lässt sich hinsichtlich der genutzten Ressourcen testen.

Die folgende Tabelle 3.4 bündelt zusammenfassend alle nicht funktionalen Anforderungen und schließt damit das Kapitel Analyse ab:

| | |
|---|---|
| | **Zuverlässigkeit** |
| **NFA1:** | Der Server, auf dem der Webshop gehostet wird, garantiert eine Verfügbarkeit von mindestens 99,99%. |
| **NFA2:** | Das System verfügt über keine Single Points of Failure. |
| **NFA3:** | Es existiert ein Wiederherstellungsmechanismus, der nach einem Ausfall das System erneut aufbauen kann. |
| | **Effizienz** |
| **NFA4:** | Innerhalb von maximal drei Sekunden können passende Empfehlungen und gezielte Angebote analysiert und dem Kunden bereitgestellt werden. |
| **NFA5:** | Zur gleichen Zeit können Daten von 1000 Kunden, die im Durchschnitt drei Klicks pro Minute tätigen, verarbeitet werden. |
| **NFA6:** | Ein kontinuierlich eintreffender Datenstrom kann mit den vorhandenen Ressourcen verarbeitet werden. |
| | **Wartbarkeit** |
| **NFA7:** | Die einzelnen Komponenten der Anwendung sind unabhängig voneinander modifizier- und erweiterbar. |
| **NFA8:** | Das System lässt sich bezüglich seiner Funktionalitäten testen. |
| **NFA9:** | Das System lässt sich hinsichtlich der genutzten Ressourcen testen. |

Tabelle 3.4: Nicht-Funktionale Anforderungen

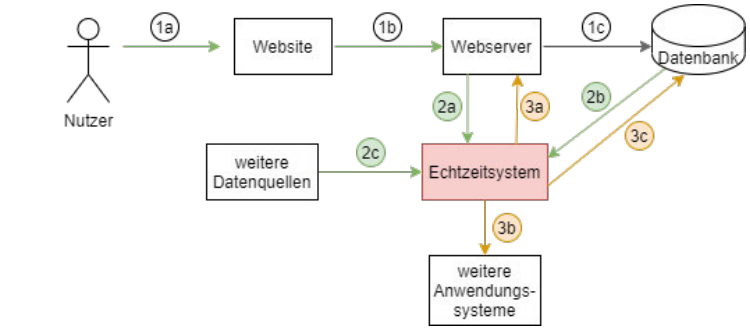
# 4 Konzeption

Das Kapitel Konzeption soll als Grundlage für die anschließende Umsetzung dienen. Hierbei liegen die in Kapitel 3.3 gestellten funktionalen sowie nicht funktionalen Anforderungen im Fokus.

Zunächst soll ein allgemeines Verständnis durch eine Beschreibung des Systemkontextes gewonnen werden, darauffolgend wird die fachliche Systemarchitektur vorgestellt, die den Schwerpunkt auf die zur Verfügung stehenden Daten sowie die Definition der Angebots- und Empfehlungslogik legt. Abschließend wird auf die einzelnen Komponenten des Prototypen sowie auf ihre Schnittstellen eingegangen.

## 4.1 Systemkontext

Das zu entwickelnde System hat die Aufgabe, unter der Berücksichtigung von eintreffenden Datenströmen und gespeicherten Stammdaten, personalisierte Angebote und Empfehlungen Kunden in Echtzeit bereit zu stellen. Als zentraler Baustein gilt das Echtzeitsystem. In dieses fließen die Kundeninteraktionen, zur Analyse benötigte Stammdaten aus einer Datenbank sowie Ereignisse aus weiteren Datenquellen (Abb. 4.1 grüne Pfeile -> Input). Im Echtzeitsystem werden die Inputdaten verarbeitet und nachfolgend ermittelte Angebote und Empfehlungen dem Kunden präsentiert, weiteren Anwendungssystemen zur Verfügung gestellt und für spätere Analysen relevante Ergebnisse in der Datenbank gespeichert (Abb. 4.1 orange Pfeile -> Output).




Legende der Abbildung:

1a) Nutzer besucht Webshop
1b) Website tauscht Daten mit Webserver aus
1c) Webserver liest Daten zur Bereitstellung der Website aus Datenbank
2a) Nutzer erzeugt Traffic Daten, die über den Webserver das Echtzeitsystem erreichen
2b) Erzeugte, modifizierte und gelöschte Stammdaten erreichen das Echtzeitsystem über die Datenbank
2c) Weitere Ereignisse von zusätzlichen Datenquellen wie Bestände im Lager oder aktuelle Angebote der Konkurrenz erreichen das Echtzeitsystem
3a) Das Echtzeitsystem schickt die ermittelten personalisierten Angebote und Empfehlungen über den Webserver zurück an den Nutzer
3b) Das Echtzeitsystem schickt die Analyseergebnisse an nachgelagerte Anwendungssysteme
3c) Die Analyseergebnisse werden in der Datenbank gespeichert und stehen für weitere Analysen bereit

Abbildung 4.1: Systemkontext anhand eines Beispielnutzers (eigene Darstellung)

Zur Vereinfachung wird für das zu entwickelnde System keine Oberfläche entwickelt. Anstelle dessen erreichen das Echtzeitsystem lediglich simulierte Kunden-Klicks. Die Ausgabe der ermittelten Angebote und Empfehlungen je Kunde erfolgt im Terminal. Außerdem wird auf nachgelagerte Anwendungssysteme verzichtet (Abb. 4.1 3b) und die Analyseergebnisse werden nicht in der Datenbank gespeichert (Abb. 4.1 3c).

## 4.2 Fachliche Systemarchitektur

Im Anschluss an die allgemeine Beschreibung des Kontextes soll mit Hilfe der funktionalen Anforderungen eine fachliche Systemarchitektur entwickelt werden. Die Architektur wird entsprechend der in den Grundlagen vorgestellten Event-Driven Architecture (siehe Kapitel 2.2.3.1) aufgebaut, bei der die Ereignisse im Mittelpunkt der Betrachtung stehen. Angelehnt an die Abbildung 2.9 sieht die Architektur des Systems wie folgt aus:

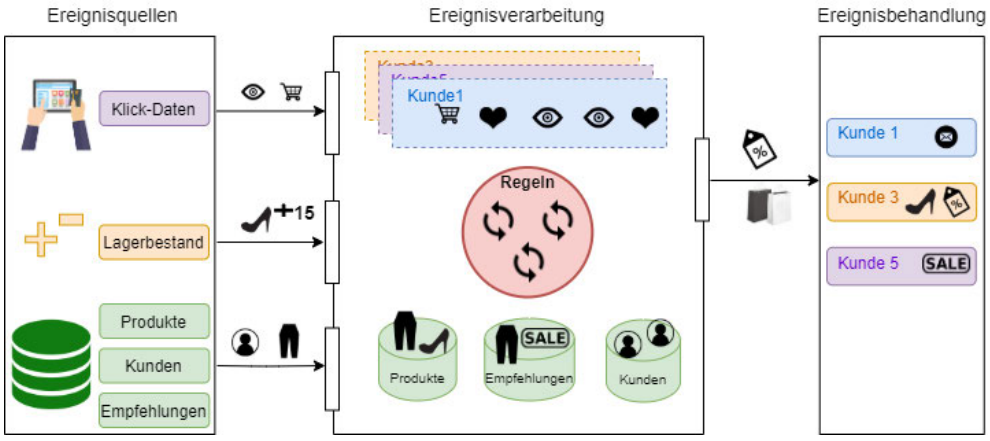



Abbildung 4.2: EDA mit fachlicher Semantik (angelehnt an [22, S.15])

### 4.2.1 Ereignisquellen

| Relevant | | Funktionale Anforderung |
|---|---|---|
| | | **Datenverarbeitung in Echtzeit** |
| ✓ | **FA1:** | Kundeninteraktionen müssen in Echtzeit nach Verhaltensmustern untersucht werden. |
| | | **Datenbankanbindungen** |
| ✓ | **FA3:** | Detaillierte und aktuelle Kundendaten stehen dem System zur Verfügung. |
| ✓ | **FA4:** | Standardisierte und aktuelle Produktdaten stehen dem System zur Verfügung. |
| | | **Angebotsentwicklung** |
| ✓ | **FA5:** | Auf Basis der Produktdaten lassen sich Empfehlungen ableiten. |
| | | **Anbindung weiterer Datenströme** |
| | **FA10:** | Ein Datenstrom externer Datenquellen kann integriert und verarbeitet werden. |
| ✓ | **FA11:** | Ein Datenstrom interner Datenquellen kann integriert und verarbeitet werden. |

Tabelle 4.1: Funktionale Anforderungen: Ereignisquellen

Aus den funktionalen Anforderungen lassen sich entsprechende Ereignisquellen ableiten. So fordert **FA1**, dass die Kundeninteraktionen - in der Abbildung 4.2 als Klick-Daten übernommen – in Echtzeit das System erreichen. Des Weiteren sollen dem System - nach **FA3** und **FA4** - aktuelle Kunden- sowie Produktdaten zur Verfügung stehen. Auch sollen zusätzliche Datenströme angebunden werden. Der Prototyp wird zur Vereinfachung nur den internen Datenstrom von Lagerbestandsveränderungen berücksichtigen (**FA11**). In **FA10** geforderte externe Datenströme - wie z.B. ein Datenstrom von Produktpreisen konkurrierender Anbieter – werden vernachlässigt. **FA5** soll insofern umgesetzt werden, dass Empfehlungen pro Produkt im Vorfeld ermittelt werden und dem Echtzeitsystem als Ereignisquelle zugänglich gemacht werden.

Demnach fließen als Ereignisquellen die Kundeninteraktionen im Webshop, Veränderungen der Stammdaten und Empfehlungen in der Datenbank sowie sich verändernde Lagerbestände in das Echtzeitsystem ein. Dass bedeutet sobald sich bspw. ein neuer Kunde registriert oder ein bestehender Kunde seine Adresse verändert, werden die Änderungen in der Datenbank erkannt und weitergeleitet. Gleiches gilt bei einer Neuaufnahme eines zu verkaufenden Produktes oder bei Veränderungen von Produkteigenschaften. Während die Klick-Daten der Kunden sowie die sich ändernden Lagerbestände mit einer hohen Frequenz das System erreichen, treffen Ereignisse, die sich aus den Veränderungen von Stammdaten ergeben, mit einer deutlich geringeren Frequenz ein. Die Strukturen der eintreffenden Klick-Ereignisse, der Lagerveränderungen und Empfehlungen sowie der Aufbau der Kunden- und Produktdaten werden im folgenden Kapitel 4.3 vorgestellt.

### 4.2.2 Ereignisverarbeitung

Wie aus der Abbildung 4.2 hervorgeht und unter 4.2.1 angesprochen wurde erreichen die Stammdaten und die Empfehlungen die Ereignisverarbeitung als Ereignisstrom. Dadurch kann gewährleistet werden, dass zur Anreicherung der Klick-Daten stets eine aktuelle Datenbasis verwendet wird. Um auf die Stammdaten durchgängig und schnell zugreifen zu können, werden die Kunden- und Produktdaten sowie die Empfehlungen im Echtzeitsystem gespeichert. Eine alternative Herangehensweise, die Stammdaten zu erhalten, wäre eine direkte Abfrage an die Datenbank pro Ereignis, wobei einzelne Abfragen für jedes Ereignis nicht effizient für eine Echtzeitanalyse wären und die Datenbank nicht für unzählige Anfragen pro Minute ausgelegt ist. Außerdem wäre eine einmalige Kopie der gesamten Datenbank in den Arbeitsspeicher möglich, hierbei wäre die Datenbasis allerdings nach der ersten Modifikation der Stammdaten nicht mehr aktuell.

| Relevant | | Funktionale Anforderung |
|---|---|---|
| | | **Datenverarbeitung in Echtzeit** |
| ✓ | **FA2:** | Kunden müssen identifiziert und ihre Interaktionen unabhängig voneinander verarbeitet werden. |
| | | **Angebotsentwicklung** |
| ✓ | **FA6:** | Die Angebots- und Empfehlungslogik lässt sich modifizieren und erweitern. |
| (✓) | **FA7:** | Kunden können anhand ihrer Eigenschaften und ihres Verhaltens segmentiert werden und entsprechend passende Angebote erhalten. |
| ✓ | **FA8:** | Ein Kunde erhält eine begrenzte Anzahl an Angeboten pro Session. |

Tabelle 4.2: Funktionale Anforderungen: Ereignisverarbeitung

Um **FA2** zu erfüllen ist es notwendig, dass das Echtzeitsystem die eintreffenden Kundeninteraktionen kundenindividuell nach Mustern durchsuchen kann. Die zu suchenden Muster sind dabei in Regeln verfasst, die sich aus der definierten Angebots- und Empfehlungslogik ergeben. Die in Kapitel 2.2.3.3 angesprochenen leichtgewichtigen CEP Komponenten eines Echtzeitsystems erleichtern notwendige Anpassungen oder Erweiterungen der Logik und realisieren damit **FA6**. Eine Kunden-Clusterung - wie in Kapitel 2.1.5 vermittelt - wird für den Prototypen nicht vorgesehen, da dies über den Umfang der Arbeit hinausreichen würde. Allerdings sollen einzelne Angebote nur bestimmten Kundengruppen vorgeschlagen werden. Die Filterung der Kunden übernehmen dabei die definierten Regeln. Somit findet in gewissen Maßen eine in **FA7** geforderten Segmentierung der Kunden statt. Zuletzt soll die Anzahl von Angeboten und Empfehlungen an Kunden durch anschließende Filterungen reduziert werden und somit **FA8** abdecken. Die formulierten Angebots- und Empfehlungs-Regeln werden im Kapitel 4.4 aufgelistet.

### 4.2.3 Ereignisbehandlung

| Relevant | | Funktionale Anforderung |
|---|---|---|
| | | **Schnittstellen zu nachgelagerten Anwendungen** |
| ✓ | **FA9:** | Empfehlungen und Angebote werden Kunden bereitgestellt. |

Tabelle 4.3: Funktionale Anforderungen: Ereignisbehandlung

Die Ereignisbehandlungsschicht ist schlicht gehalten. Der Prototyp umfasst weder eine nachgelagerte Kunden-Clusterung noch folgt ein entwickeltes Empfehlungssystem. Lediglich die in FA9 geforderte Anforderung soll vereinfacht dargestellt werden. Angebote und Empfehlungen sollen je Kunde als String ausgegeben werden und könnten theoretisch somit im Anschluss von Webservern, die den Kunden informieren, sowie anschließenden Anwendungssystemen verwendet und weiterverarbeitet werden.

Struktur einer beispielhaften Ausgabe: Timestamp, Kunden-ID, Typ der Personalisierung, Mitteilung mit persönlichem Angebot
Eine beispielhafte Ausgabe: 02-16-2020 14:33:45, 1307, Angebot, Melde Dich für den Newsletter an und sichere Dir 10% Rabatt auf deinen nächsten Einkauf!

## 4.3 Datenstruktur

Im Folgenden werden die Strukturen der Klick-Ereignisse, der Lagerbestände, der Empfehlungen sowie der Kunden- und Produkt-Stammdaten festgelegt. Auf Grund der Tatsache, dass keine Beispieldatensätze zum Kunden-Traffic vorliegen, gilt es, diese in der Umsetzung zweckmäßig zu generieren.

**Kundeninteraktionen**


Das eintreffende Klick-Ereignis des Kunden setzt sich aus einem Timestamp, einem Ereignistypen, der Page-URL sowie der Kunde-ID zusammen. Ereignistypen werden auf das Anschauen, Merken sowie Kaufen eines Produktes reduziert. Dabei soll der Typ Anschauen am häufigsten und der Typ Kaufen am seltensten vorkommen. Aus der Page-URL ergibt sich das jeweilige Produkt und mit Hilfe der Kunden-ID ist es möglich den entsprechenden Kunden zu identifizieren.

Zur Vereinfachung wird davon ausgegangen, dass es sich bei allen Nutzern des Webshops um registrierte und eingeloggte Kunden handelt. Eine Beispiel Page-URL stellt sich wie folgt dar: https://www.personalisierterShop.de/Schuhe/Adidas/produktID=11944

Weitere in der Analyse angesprochenen Attribute wie die IP-Adresse, eine vorherige URL sowie technische Details werden vernachlässigt, da sie für die Analyse keinen gesonderten Mehrwert bieten.

### Lagerbestände

| *Lagerbestand* |
| --- |
| Timestamp |
| Produkt_ID |
| Bestand |

Die im Datenstrom eintreffenden Lagerbestände sind sehr einfach gehalten. Sie sollen ausschließlich aus einem Timestamp, aus einer Produkt-ID zur Identifikation des Produktes und der aktuellen veränderten Bestandsmenge bestehen.

### Kundendaten

| *Kunde* |
| --- |
| Kunden_ID |
| Vorname |
| Nachname |
| Geschlecht |
| Geburtsdatum |
| Straße |
| Haus-Nr. |
| Stadt |
| Telefon |
| E-Mail |
| Newsletter |
| Registrierungsdatum |

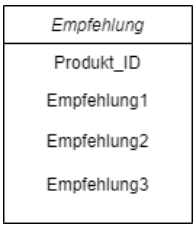
Die Kundendaten beinhalten die Kunden-ID, den Namen, das Geschlecht, das Geburtsdatum, die Adresse, eine E-Mail-Adresse und Telefonnummer sowie die Zustimmung oder Ablehnung des Newsletter Abbos und das Registrierungsdatum. Aus dem Registrierungsdatum lässt sich der Kundentyp ermitteln. Bevor der Kunde als Bestandskunde gilt, gilt er im ersten Jahr als Neukunde.
Daten, die sich aus älteren Besuchen ergeben würden und somit nach einer Session in der Datenbank hinterlegt werden müssten, wie die Bestellhistorie, in einer Wunschliste gespeicherte Produkte, präferierte Marken oder Größen, werden nicht in die Auswahl der Angebote und Empfehlungen mit einbezogen.

### Produktdaten

| *Produkt* |
| --- |
| Produkt_ID |
| Produktname |
| Marke |
| Preis |
| Farbe |
| Kategorie |
| Subkategorie |
| Geschlecht |
| Verkaufsorganisation |

Produktdaten bestehen aus der Produkt-ID, dem Produktnamen, einer Marke, einem Preis, einer Farbe, einer Kategorie sowie Subkategorie. Des Weiteren lassen sie sich einem Geschlecht zuordnen und enthalten eine Verkaufsorganisation, die angibt, ob das Produkt neu ist oder reduziert ist oder aber als Standardprodukt angeboten wird. Produktbewertungen sowie Größen werden bei der Analyse vernachlässigt.
Immer wenn es zu Veränderungen in der Kunden- oder Produkt-Tabelle kommt, werden die veränderten Daten als Ereignisse an das Echtzeitsystem geschickt.

**Empfehlungen**




Empfehlungen werden im Vorfeld auf Grund von den in der Tabelle 4.5 definierten Empfehlungsregeln je relevantes Produkt bestimmt. Eine Empfehlung besteht aus vier Produkt-IDs. Die erste Produkt-ID entspricht dem Produkt, zudem Empfehlungen vorgeschlagen werden sollen. Die drei weiteren IDs ergeben die Empfehlungen zur ersten Produkt-ID.
Genau wie die Kunden- und Produktdaten erreichen neue oder veränderte Empfehlungen das Echtzeitsystem als Ereignisstrom.

## 4.4 Angebots- und Empfehlungslogik

Mittels der zur Verfügung stehenden Daten sollen in diesem Kapitel die Regeln für die Personalisierung festgelegt werden. Bevor die Regeln zur Mustererkennung relevant werden, werden die eintreffenden Ereignisse zunächst aufbereitet. Mit Hilfe der in die Ereignisverarbeitung geladenen Stammdaten können die Klick-Ereignisse um die Kunden- sowie Produktdaten angereichert werden. Eine Zuordnung ist durch die Kunden-ID und die aus der Page-URL ableitbaren Produkt-ID möglich.

Sind die Ereignisse aufbereitet, beginnt die Mustererkennung. Die dafür benötigte Wissensbasis ergibt sich aus der Zusammensetzung von Regeln zu personalisierten Angeboten und Hinweisen sowie personalisierten Produktempfehlungen.

| | |
|---|---|
| **A1:** | Wenn ein Kunde den Newsletter noch nicht abonniert hat, dann weise ihn auf eine Preisreduzierung um 10% bei entsprechendem Abonnement hin. |
| **A2:** | Wenn ein Kunde als Neukunde gilt oder unter 25 Jahre alt ist und sich ein Produkt merkt, dann gewähre ihm einen kostenlosen Versand. |
| **A3:** | Wenn sich ein Kunde ein Produkt fünf Mal anschaut oder merkt, dann gewähre ihm einen Rabatt-Code von 10% auf das spezielle Produkt. |
| **A4:** | Wenn ein Kunde sich innerhalb von zehn Klicks mind. sechs Produkte im Sale oder sich mind. sechs neue Produkte angeschaut oder gemerkt hat, dann sende ihm einen 10% Rabatt-Code auf die jeweilige Verkaufsorganisation. |
| **A5:** | Wenn ein Kunde sich ein Produkt angeschaut oder gemerkt hat, dann weise ihn bei geringer Verfügbarkeit darauf hin. Unter einer geringen Verfügbarkeit ist ein Bestand unter fünf zu verstehen. |

Tabelle 4.4: Personalisierte Angebote und Hinweise

Damit Angebote für die Kunden nicht zur Selbstverständlichkeit werden, wird sichergestellt, dass je Angebotstyp maximal ein Angebot pro Stunde vorgeschlagen wird.

Die Regeln zu Empfehlungen werden ausschließlich auf Basis der zur Verfügung stehenden Daten ermittelt. Es werden keine Algorithmen von reellen Empfehlungssystemen verwendet, da dies den Umfang der Thesis übersteigen würde.

Auf folgenden Regeln basieren die Empfehlungen an Kunden:

| | |
|---|---|
| **E1:** | Wenn ein Kunde innerhalb von zwei Minuten mind. zu 50% Produkte der gleichen Kategorie angeklickt hat, dann empfehle ihm drei Produkte aus der Kategorie, die zum Geschlecht des zuletzt betrachteten Produktes der Kategorie passen. |
| **E2:** | Wenn ein Kunde unter 30 Jahre alt ist und innerhalb von zwei Minuten zu mind. 50% Produkte unter 50€ angeklickt hat, dann empfehle ihm drei farbenfrohe Produkte unter 50€, die zum Geschlecht des zuletzt betrachteten Produktes unter 50€ passen. |
| **E3:** | Wenn ein Kunde innerhalb von zwei Minuten zu mind. 70% Produkte über oder gleich 50€ angeklickt hat, dann empfehle ihm drei Produkte über 50€, die sich nicht im Sale befinden und zum Geschlecht des zuletzt betrachteten Produktes über oder gleich 50€ passen. |

Tabelle 4.5: Personalisierte Empfehlungen

Die verschiedenen Regeln können beliebig ergänzt und erweitert werden. Für diese Arbeit sollen sie aber genügen, da sie die Ziele der Personalisierung im E-Commerce abdecken und weitere Regeln keinen konkreten Mehrwert der Arbeit bieten würden.

## 4.5 Streaming Analytics Tools

Bevor im Anschluss an die fachliche Perspektive des Systems die Komponenten und ihre Verbindungen thematisiert werden, soll vorab das im Mittelpunkt stehende Analysetool zur Echtzeitverarbeitung ausgewählt werden.

Die Prüfung eines geeigneten Echtzeittools konzentriert sich auf die weit verbreiteten Open Source Frameworks Apache Spark, Apache Storm sowie Apache Flink. Bei der Entscheidung sollen insbesondere die Nicht-Funktionalen Anforderungen (Tabelle 3.4) zur

Effizienz unterstützen. Vorab werden zwei grundlegende Aspekte, die eine Verarbeitung mit kurzer Latenzzeit ermöglichen, vorgestellt.

**Datenbankmanagementsysteme (DBMS)**
Traditionelle DBMS speichern ihre Daten auf einer Festplatte und verschieben diese bei einem Zugriff in einen Cache im Hauptspeicher, um bei erneutem Zugriff die Effizienz zu erhöhen. Das ständige Verschieben der Daten trägt jedoch zu einer schlechteren Performance bei, die für eine Echtzeitanwendung nicht ausreichend ist. Aus diesem Grund wurden In-Memory-Datenbankmanagementsysteme entwickelt, die ihre Datenspeicherung, Verwaltung sowie Manipulation überwiegend im Hauptspeicher durchführen. Die Latenzzeit kann gemindert, der Befehlssatz, der für den Zugriff auf die Daten zuständig ist, reduziert und schlussendlich die Performance von Anwendungen und Abfragen verbessert werden. Allerdings müssen die Daten regelmäßig in einen persistenten Speicher verschoben werden, da sie im Hauptspeicher bei einem Ausfall verloren gehen würden.

**Data Streaming**
Neben einem In-Memory-DBMS zeichnet ein Echtzeittool die Verarbeitung von Datenströmen aus, wie es beim Stream Processing vorgesehen ist. Sobald Daten erzeugt bzw. empfangen werden, findet die Verarbeitung statt. Stream-Prozessoren empfangen, verarbeiten und senden die Ereignisse. Dabei sind sie dafür verantwortlich, dass der Datenfluss sowie die Verarbeitung effizient und fehlertolerant sind. Latenzzeiten sollten je Fall Sekunden bzw. Millisekunden nicht überschreiten. Schrittweise werden Berichte und Metriken abhängig von den eintreffenden Daten aktualisiert. Das Gegenstück zur Stream-Verarbeitung ist die Batch-Verarbeitung. Sie ermöglicht Analysen über große Datensätze. Im Gegensatz zum Stream-Processing berechnet das Batch-Processing in der Regel Ergebnisse aus einer großen Menge gespeicherter Daten, die Latenzen von Minuten bis Stunden haben können [58, S.76ff.].

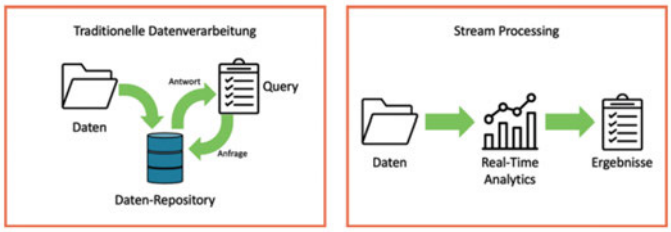



Abbildung 4.3: Traditionelle Datenverarbeitung vs. Stream Processing [58, S.77]

### 4.5.1 Apache Spark

Bei Apache Spark handelt es sich um ein Open Source Framework, das 2009 an der University of California in Berkeley entwickelt wurde. Der Fokus der Entwicklung lag auf der Beschleunigung von Batchverarbeitung-Workloads, die durch eine vollständige In-Memory-Verarbeitung und Prozessoptimierungen erfolgt. Durch die In-Memory-Verarbeitung können große Datenmengen 100-mal schneller verarbeitet werden als in Hadoop. Apache Spark ist ein hybrides Verarbeitungssystem. Es ist sowohl in der Lage Batch- als auch Stream-Verarbeitung auszuführen. Darüber hinaus sind zahleiche Operatoren integriert und es verfügt über ein Bibliothekssystem, das u.a. maschinelles Lernen, SQL-Abfragen und Diagrammverarbeitung unterstützt [25].

Spark wird in der Literatur auch als Near to Real-Time Framework bezeichnet, da es sich bei der Verarbeitung der Daten um keine echte Streaming Verarbeitung handelt [3]. Grundsätzlich verarbeitet die Engine die Daten mit Hilfe des Batch-Processings. Um jedoch auch in Echtzeit Analysen durchführen zu können, wurde das Konzept mit dem Namen Micro-Batch Processing erarbeitet. Datenströme werden in sehr kleine Stapel von Millisekunden unterteilt, die dann als kleine feste Datensätze von der Engine verarbeitet werden können [12].

Abschließend lässt sich festhalten, dass Apache Spark eine gute Wahl ist, wenn unterschiedliche Verarbeitungsaufgaben relevant sind. Vor allem die Batch-Verarbeitung überzeugt durch hohe Geschwindigkeiten im Vergleich zu Hadoop. Im Vergleich zu reinen Stream-Verarbeitungssystemen zeigt die Spark-Stream-Verarbeitung durch das Micro-Batch Processing eine höhere Latenz. Sind der Durchsatz und die Latenz jedoch gleichermaßen wichtig, ist auch Spark Streaming eine gute Option für die Entwicklung eines Echtzeitsystems.

### 4.5.2 Apache Storm

Das Open Source Framework Apache Storm wurde zentral für die Verarbeitung von Streaming Daten in Echtzeit entwickelt und unterstützt somit ausschließlich das Stream-Processing. Es zeichnet sich durch seine hohe Geschwindigkeit bei der Verarbeitung von großen Datenmengen aus und findet daher insbesondere in Programmen Anwendung, die eine sehr niedrige Latenz fordern. In einem Cluster können pro Sekunde und Knoten über eine Million Datensätze verarbeitet werden [3].

Die Storm Stream-Verarbeitung erhält von den Stream-Quellen (Spouts) die kontinuierlich eintreffenden Datenströme. Die sogenannten Bolts führen die definierten Funktionen aus, filtern, aggregieren oder verknüpfen die Daten und kommunizieren mit Datenbanken. Sie sind demnach für die Verarbeitung des Eingangs-Streams und die Generierung des Ausgangs-Streams zuständig. Die Netzwerkdarstellung der gesamten Bolts sowie Spouts wird als Topologie bezeichnet. Die Idee hinter Apache Storm ist das Definieren von kleinen Operationen, die dann zu einer Topologie zusammengesetzt werden können [24]. Es wird garantiert, dass jede Dateneinheit mindestens einmal verarbeitet wird, d.h. aber auch dass Fehlerszenarien zu Duplikaten führen können.

Es lässt sich festhalten, dass Apache Storm sehr gut für Anwendungen geeignet ist, die sehr hohe Latenzanforderungen stellen und für die einzig das Stream-Processing relevant ist. Auch Storm bietet eine umfassende Sprachunterstützung und viele Optionen zum Definieren von Topologien an [13].

### 4.5.3 Apache Flink

Das 2010 von drei deutschen Universitäten entwickelte Apache Flink ist wie Apache Spark und Apache Strom ein Open Source Framework. Mit Hilfe einer Stream- und Batch-Verarbeitungs-API kann sowohl die Stream- als auch die Batch-Verarbeitung durchgeführt werden. Im Unterschied zu Spark ist Flink ein Real-Time Framework mit richtigem Stream-Processing, d.h. eintreffende Daten werden Element für Element als echter Stream verarbeitet, so wie es auch bei Storm der Fall ist [39]. Das Batch-Processing wird durch die Betrachtung von Datenströmen mit endlichen Grenzen ermöglicht.

Da es sich bei Flink ebenfalls um eine In-Memory-Verarbeitung handelt, wird die Persistenz der Daten durch Momentaufnahmen zu festgelegten Zeitpunkten gewährleitet. Tritt ein Fehler auf, werden die Quellen zurückgespult, der Status wiederhergestellt und die Verarbeitung kann fortgesetzt werden. So gelingt es eine fehlertolerante und genau einmalige Semantik bereitzustellen. Auch Flink bietet unterstützende Libraries zu SQL-Abfragen, maschinellem Lernen sowie zur Graph-Verarbeitung. Des Weiteren lässt es sich lokal, in einem Cluster oder in einer Cloud deployen und ist in der Lage Daten parallel zu verarbeiten [9].

Schlussendlich bietet Apache Flink sowohl eine Stream-Verarbeitung mit geringer Latenz als auch eine Unterstützung für Batch-Aufgaben. Sein Stream-First-Ansatz sorgt für eine echte Verarbeitung von Element zu Element und erreicht somit eine bessere Latenzzeit

als Apache Spark. Außerdem kann Flink mit einem hohen Durchsatz überzeugen, der als deutlich performanter als bei Apache Storm gilt [39]. Auch die Dokumentation ist überzeugend und klar strukturiert [9].

### 4.5.4 Auswahl des Tools

Aus der Vorstellung der einzelnen Frameworks zeigt sich, dass alle drei für eine Echtzeitanalyse geeignet sind und jeweils ihre Vorzüge mit sich bringen. Für das zu entwickelnde System fiel die Entscheidung auf Apache Flink, da eine Entwicklung mit einer echten Streaming Verarbeitung von Interesse ist und die Community um Flink stetig wächst. Der größte E-Commerce Anbieter Chinas Alibaba sowie weitere bekannte Unternehmen wie Uber und Zalando setzen bei der Umsetzung von Echtzeitsystemen bereits auf Apache Flink [9]. Darüber hinaus ergeben weitere Analysen, dass sich Flink in der Streaming Verarbeitung behaupten kann. Beispielsweise kommen die Autoren des Papers „A Comparative Study of Big Data Frameworks“ [3] zu dem Ergebnis, dass sich Flink gegenüber Spark und Storm in Bezug auf die Verarbeitungszeit, den CPU-Verbrauch, die Latenzzeit, den Durchsatz sowie der Ausführungszeit dursetzen kann. Dieses Ergebnis ermitteln die Autoren aus zahlreichen Studien, die die Frameworks zuvor auf bestimmte KPIs miteinander verglichen haben.

## 4.6 Systemkomponenten

Nachdem die Entscheidung für ein Echtzeittool getroffen wurde, werden im Folgenden die weiteren Komponenten sowie ihre Schnittstellen beschrieben. Abbildung 4.4 gibt einen Überblick der für die Anwendung benötigten Komponenten.

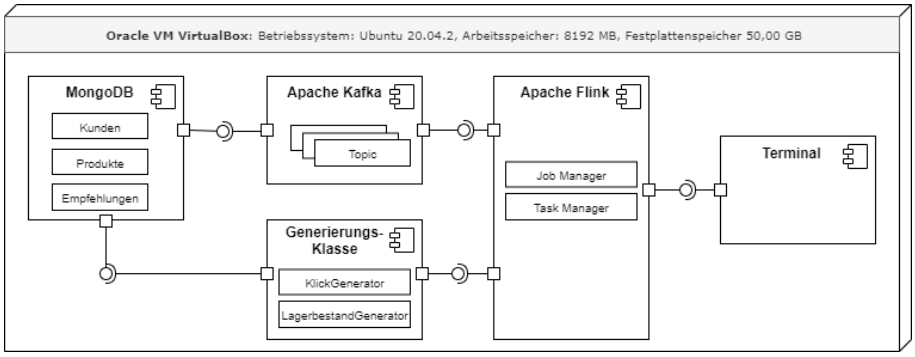



Abbildung 4.4: Systemkomponenten (eigene Darstellung)

### 4.6.1 MongoDB

Die Stammdaten und Empfehlungen befinden sich in der MongoDB. Die MongoDB gilt als eine beliebte, kostenfreie, dokumentenbasierte Datenbank für NoSQL. In den Mittelpunkt der Speicherung rücken die Dokumente, die keine vorgegebenen Strukturen einhalten müssen. Hierdurch können neue Informationen einfach in einem Dokument ergänzt werden, ohne dass alle Dokumente bearbeitet werden müssen. Trotzdem ist es für eine effiziente Bearbeitung sinnvoll, Dokumente einer ähnlichen Struktur zu verwenden [38, S.299-304]. Als zentrale Datenstruktur stellt die MongoDB Collections von JSON-Dokumenten zur Verfügung. Darüber hinaus ermöglicht eine ausdrucksreiche Abfragesprache die Filterung sowie Sortierung nach beliebigen Feldern und die Unterstützung von Aggregationen [43].

### 4.6.2 Apache Kafka

Veränderungen der Stammdaten in der MongoDB werden von Apache Kafka erkannt und für Apache Flink zugängig gemacht. Apache Kafka bietet sich als ein skalierbares Messaging System für die Verbindung von der MongoDB zu Apache Flink an. Kafka gilt als universelle Schnittstelle zwischen Systemen, die für die Datenbereitstellung zuständig sind und zeichnet sich durch eine besonders hohe Skalierbarkeit aus. In einer Art Buffer werden die Daten in sogenannten Topics zwischengespeichert. Dabei sind Producer für das Befüllen und Consumer für das Auslesen der Topics zuständig. Beim Abarbeiten von Nachrichten können die Datenströme partitioniert und für eine höhere Ausfallsicherheit repliziert werden [47, S.449-456].

Neben den Consumern und Producern existieren eine Streams- sowie eine Connect-API. Unter der Verwendung der Connect-API lassen sich Producer und Consumer mit bestehenden Systemen und Speichersystemen verbinden [11]. Sowohl Apache Flink als auch die MongoDB verfügen über so einen Connector und lassen sich daher unkompliziert durch Apache Kafka verbinden [42].

### 4.6.3 Weitere Komponenten

**Generierungs-Klasse**
Die Generierungs-Klasse ist für die Simulation von Kundeninteraktionen und Veränderung von Lagerbeständen zuständig. Sie baut eine Verbindung zur MongoDB auf und fragt für die Generierung benötigte Kunden- sowie Produktdaten ab. Generierte Kundeninteraktionen und Lagerbestände erreichen Apache Flink über eine Client-Server Kommunikation mittels Sockets.

**Apache Flink**
Die Verarbeitung von Ereignissen in Echtzeit wird durch das ausgewählte Framework Apache Flink ermöglicht. Unter Verwendung des Arbeitsspeichers sowie durch die integrierte DataStream API gelingt es Apache Flink Datenströme effizient zu verarbeiten. Außerdem bietet das Framework einige vordefinierte Operationen, die die Entwicklung erleichtern und den Programmieraufwand reduzieren. Zur Verfügung stehende Konnektoren stellen die Schnittstellen zur Generierungs-Klasse, zu Apache Kafka und zum Terminal bereit.

Apache Flink unterstützt eine zustandsorientierte Stream-Verarbeitung, die das Speichern von Informationen über mehrere Ereignisse hinweg zulässt. Dies ist notwendig, um Anwendungen nach bestimmten Ereignismustern durchsuchen und auf historisierte Ereignisse effizient zugreifen zu können.

Das Framework wird als Standalone Cluster betrieben. Dieser besteht mindestens aus einem Client, einem Job Manager sowie einem Task Manager. Der Client nimmt dabei den Programmcode der Flink-Anwendung entgegen und wandelt diesen in einen Job-Graphen um, sodass er dem Job Manager zur Verfügung gestellt werden kann. Der Job Manager koordiniert die Arbeit und verteilt sie an die Task Manager. Diese führen schließlich die Operationen eines Flink-Jobs aus. Für den Prototypen soll es genügen lediglich einen Task Manager bereitzustellen [9].

**Terminal**
Die Ergebnisse der Analyse werden im Terminal ausgegeben.

### 4.6.4 Thematisierung Nicht-Funktionaler Anforderungen

Nach der Beschreibung der Komponenten soll überprüft werden, ob und wie die nicht funktionalen Anforderungen umgesetzt werden können. Der Schwerpunkt fällt bei der Prüfung auf das ausgewählte Echtzeittool, das zentral für die Verarbeitung der Daten zuständig ist.

**Zuverlässigkeit**
NFA1 kann erfüllt werden indem die gesamte Anwendung auf eine hochverfügbare Cloud deployed wird. Beispielsweise verspricht der Cloud-Computing-Anbieter Amazon Web Services (AWS) [7] eine mindest Verfügbarkeit von 99,9% für seinen Amazon EMR Service, der Flink als YARN-Anwendung unterstützt. Auch die Amazon Services MSK und DocumentDB versprechen einen vollständig verwalteten, hochverfügbaren und sicheren Apache Kafka-Service sowie einen schnell skalierbaren, hochverfügbaren MongoDB kompatiblen Datenbankservice.

Um in Apache Flink Single Points of Failure zu eliminieren, ist es u.a. entscheidend den Job Manager hochverfügbar zu gestalten. Hierfür muss ein eigenständiger Cluster mit einem einzigen führenden Job Manager sowie mehreren Standby-Job Managern laufen. Fällt der führende Job Manager aus, wird die Führung von einem Standby-Job Manager übernommen. Der Zookeeper-Dienst ermöglicht dabei die verteilte Koordination zwischen den laufenden Instanzen [9]. Bei Apache Kafka werden Single Points of Failure durch die Replikation der Topics entfernt. Ein Kafka-Cluster enthält mehrere Broker. Ein Broker ist immer Leader der Partition eines Topics. Weitere Broker im Cluster enthalten Replikationen. Fällt einer der Broker im Kafka-Cluster aus, können andere Broker die Aufgaben übernehmen und werden damit zum neuen Leader der Partition [11].

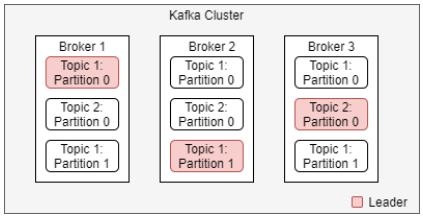



Abbildung 4.5: Kafka Cluster (eigene Darstellung)

Auch bei der MongoDB können Single Points of Failure durch Replica-Sets verhindert werden. Alle MongoDB Instanzen verwalten denselben Datensatz. Eine der Instanzen entspricht dem Primärknoten. Er empfängt alle Schreib- und Lese-Vorgänge. Aufgerufene Operationen werden auf die anderen Instanzen - die Sekundärknoten – repliziert, sodass sie im Anschluss den gleichen Datensatz enthalten wie der Primärknoten. Ist der Primärknoten nicht mehr verfügbar, übernimmt eine Sekundärinstanz die Aufgabe des Primärknotens [43]. Der MongoDB-Kafka-Source-Connector verlangt für seinen Aufbau ebenfalls ein Replica-Set [42].

Dadurch das Flink durchgehend Snapshots der verteilten Data Streams und des Transformations-Status erstellt, kann außerdem eine hohe Fehlertoleranz erzielt werden. Bricht das System ab, ist es möglich, anhand der persistierten Snapshots das System an der abgebrochenen Stelle wieder zu starten [9].

Beispielhaft soll hiermit verdeutlicht werden, dass die Anforderungen zur Zuverlässigkeit umsetzbar sind, im weiteren Verlauf der Arbeit sollen sie jedoch keine vertiefte Berücksichtigung finden. Vielmehr soll der Fokus auf der Effizienz liegen, da diese bei der Datenverarbeitung in Echtzeit von größter Bedeutung ist.

| Relevant | Nicht-Funktionale Anforderung |
|---|---|
| | **Zuverlässigkeit** |
| | **NFA1:** Der Server, auf dem der Webshop gehostet wird, garantiert eine Verfügbarkeit von mindestens 99,99%. |
| | **NFA2:** Das System verfügt über keine Single Points of Failure. |
| | **NFA3:** Es existiert ein Wiederherstellungsmechanismus, der nach einem Ausfall das System erneut aufbauen kann. |

Tabelle 4.6: Nicht-Funktionale Anforderungen: Zuverlässigkeit

**Effizienz**

Wie bereits vorgestellt handelt es sich bei Apache Flink um ein Real-Time Framework mit richtigem Stream-Processing (siehe Kapitel 4.5.3), wodurch unbegrenzte Datenströme verarbeitet werden können. Die von Flink verwendete In-Memory Verarbeitung ermöglicht auf der einen Seite eine geringe Latenzzeit, benötigt auf der anderen Seite aber eine höhere Arbeitsspeicherkapazität.

Dem Prototypen stehen folgende Ressourcen zur Verfügung:

- Betriebssystem: Ubuntu 20.04.2
- Arbeitsspeicher: 8192 MB
- Anzahl Prozessoren: 2
- Festplattenspeicher 50,00 GB

Auf der offiziellen Apache Flink Website wird über Anwendungen berichtet, die mehrere Billionen Ereignisse pro Tag verarbeiten können [9]. In Kapitel 6 sollen die konkreten Anforderungen zur Effizienz überprüft werden.

| Relevant | Nicht-Funktionale Anforderung |
|---|---|
| | **Effizienz** |
| ✓ | **NFA4:** Innerhalb von maximal drei Sekunden können passende Empfehlungen und gezielte Angebote analysiert und dem Kunden bereitgestellt werden. |
| ✓ | **NFA5:** Zur gleichen Zeit können Daten von 1000 Kunden, die im Durchschnitt drei Klicks pro Minute tätigen, verarbeitet werden. |
| ✓ | **NFA6:** Ein kontinuierlich eintreffender Datenstrom kann mit den vorhandenen Ressourcen verarbeitet werden. |

Tabelle 4.7: Nicht-Funktionale Anforderungen: Effizienz

**Wartbarkeit**

Mit einem Blick zurück in das Kapitel 2.2.3 lässt sich festhalten, dass die für die Echtzeitanalyse entwickelte Architektur eine lose Kopplung zwischen den einzelnen Komponenten besitzt. Die Ereignisquellen entsprechen der MongoDB-Kafka-Verbindung sowie der Generierungs-Klasse. Diese kennen sich weder untereinander noch wissen sie, wie ihre versendeten Ereignisse im Verlauf verarbeitet werden. Allein die Ereignisverarbeitungskomponente, hier Apache Flink, kennt die Fachlogik. Sie transformiert die eintreffenden Datenströme und erzeugt aus ihnen neue Ströme, die entweder weiterverarbeitet oder an die Ereignisbehandlungsschicht versendet werden (siehe Kapitel 4.5.3).

Dadurch, dass nur Flink die Fachlogik kennt, muss ausschließlich Flink bei Erweiterungen und Modifikationen der Logik angepasst werden. Die lose Kopplung der Quellen und Senken ermöglicht zudem Komponenten unabhängig voneinander anzupassen und zu erweitern.

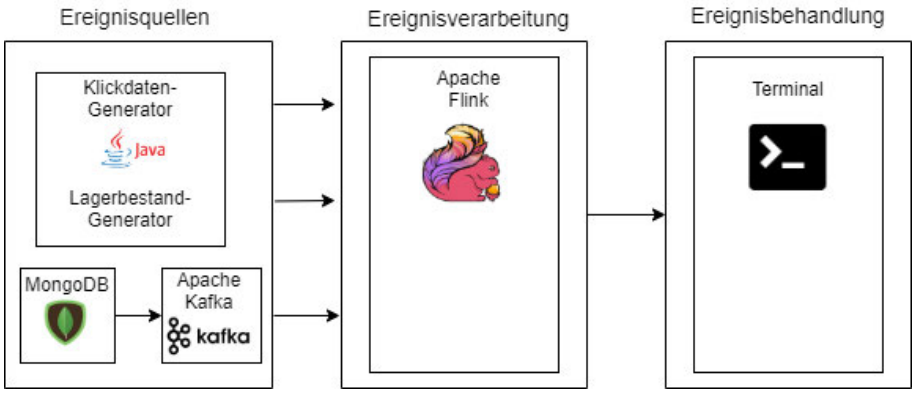



Abbildung 4.6: EDA mit den verwendeten Tools (angelehnt an [22, S.15])

Wie aus der Bewertung der Event-Driven Architecture unter Kapitel 2.2.3.5 hervorgeht, gilt ein strukturiertes Testen durch die lose Kopplung als schwierig. Außerdem erschweren eine große Menge an Regeln die Übersichtlichkeit und damit die Kontrolle des Systemverhaltens. Dennoch kann das Testen nicht vernachlässigt werden. Passend zu den definierten Angebots- und Empfehlungs-Regeln des Kapitels 4.4 werden im Kapitel Evaluation kontrollierte Inputdaten generiert und ihre Erwartungsausgabe mit der Ist-Ausgabe verglichen. Neben den fachlichen Tests soll ebenfalls die Ressourcen-Nutzung und eine effiziente Verarbeitung angesprochen werden.

| Relevant | Nicht-Funktionale Anforderung |
|---|---|
| | **Wartbarkeit** |
| ✓ | **NFA7:** Die einzelnen Komponenten der Anwendung sind unabhängig voneinander modifizier- und erweiterbar. |
| ✓ | **NFA8:** Das System lässt sich bezüglich seiner Funktionalitäten testen. |
| ✓ | **NFA9:** Das System lässt sich hinsichtlich der genutzten Ressourcen testen. |

Tabelle 4.8: Nicht-Funktionale Anforderungen: Wartbarkeit

# 5 Umsetzung

Auf Grundlage der Kapitel 3 und 4 soll das folgende Kapitel die Entwicklung der Echtzeitanwendung im Bereich Personalisierung im E-Commerce beschreiben. Im Mittelpunkt steht dabei die Ereignisverarbeitung in Echtzeit. Einleitend wird der Realisierungsumfang abgesteckt, bevor im Anschluss detailliert auf die einzelnen Bestandteile der Anwendung eingegangen wird.

## 5.1 Realisierungsumfang

Die Entwicklung kombiniert die fachlichen Aspekte und technischen Komponenten der Konzeption. So wird die fachliche Systemarchitektur (siehe Abbildung 4.2) mit den ausgewählten Tools (siehe Abbildung 4.6) umgesetzt.

Wiederholend bedeutet dies, dass sich die Ereignisquellen aus den Klick-Ereignissen der Kunden, der Veränderungen der Produktbestände sowie den Produkt- und Kundendaten als auch den Empfehlungen zusammensetzen. Während die Klick-Ereignisse und Bestandsveränderungen mit Hilfe der Kunden- und Produktdaten in einer Java-Klasse generiert werden und über eine Client-Server-Kommunikation die Ereignisverarbeitung erreichen, befinden sich die zuvor generierten Kunden- und Produktdaten sowie Empfehlungen in der MongoDB. Veränderungen in der Datenbank fließen als Datenstrom über den Mongo-Kafka-Connector in die Ereignisverarbeitung. Die Daten werden nach den in Kapitel 4.3 definierten Datenstrukturen aufgebaut.

Die Ereignisverarbeitung wird mittels des Frameworks Apache Flink umgesetzt und die in Kapitel 4.4 festgehaltene Empfehlungs- und Angebotslogik steht im Vordergrund. Resultierende personalisierte Angebote und Empfehlungen werden im Terminal symbolisch für die weitere Verarbeitung und Kundenmittelung ausgegeben.

Wie in Kapitel Konzeption erwähnt werden bei der Entwicklung des Systems die Nicht-Funktionalen Anforderungen bzgl. der Zuverlässigkeit vernachlässigt. Des Weiteren wird

auf eine Frontend Implementation verzichtet und stattdessen das Klickverhalten der Kunden simuliert. Eine klassische Clusterung der Kunden sowie ein eigenständiges Empfehlungssystem können auf Grund des Umfangs ebenfalls nicht umgesetzt werden. Mit einem Blick auf die funktionalen Anforderungen wird lediglich FA10: „Ein Datenstrom externer Datenquellen kann integriert und verarbeitet werden.“ nicht in die Entwicklung mit einbezogen.

## 5.2 Realisierungsdetails

Konkret werden im Folgenden die relevanten Details zur Umsetzung beschrieben. Nach einer kurzen Einführung in die Projektstruktur und den zur Verfügung stehenden Ressourcen wird das Kapitel nach der EDA in die Ereignisquellen, Ereignisverarbeitung und Ereignisbehandlung unterteilt. Dabei liegt der Fokus auf der Ereignisverarbeitung mit Apache Flink. Die Umsetzung dessen erfolgt auf Basis der Flink Dokumentation [9].

### 5.2.1 Hardware und Projektstruktur

Die Anwendung läuft auf einer virtuellen Maschine (VM) mit dem Betriebssystem Ubuntu 20.04.2. Der VM stehen 8GB Arbeitsspeicher sowie 50GB Festplattenspeicher und zwei Prozessoren zur Verfügung.

Ein Java-Projekt enthält den Programmcode. Dieser ist unterteilt in die Packages Entitäten und States und beinhaltet zusätzlich die Klassen Ereignisquelle, Ereignisverarbeitung sowie eine Test-Klasse.

Mit Hilfe der Klasse **Ereignisquelle** werden die Kundeninteraktionen und die sich ändernden Lagerbestände simuliert. Die Angebots- sowie Empfehlungslogik sind modifizierbar und leicht erweiterbar. Sie befinden sich in der Klasse **Ereignisverarbeitung**.

In dem Package **Entitäten** befinden sich alle relevanten Objekte. Neben dem Klick-Datum, dem Lagerbestand sowie dem Produkt, den Kunden und der Empfehlung-ID (siehe Kapitel 4.3) werden folgende Plain Old Java Objects (POJO) zur vereinfachten Umsetzung ergänzt:

- Empfehlung: Passend zur Klasse Empfehlung-ID - in der die IDs eines Produktes sowie seiner drei Empfehlungen gespeichert sind – befinden sich zu den IDs die jeweiligen Produkt-Objekte.
- Klick-Produkt-Kunde: Während das Klick-Datum ausschließlich die Produkt-ID und die Kunden-ID enthält, wird hier ebenfalls anstelle der IDs das gesamte Produkt und die vollständigen Kundeninformationen angegeben.

Das Package **States** enthält Klassen, die es ermöglichen, Daten für einen längeren Zeitraum in Flink verfügbar zu halten. So besteht die Möglichkeit, Klick-Daten mit Produkt-, Kunden- und Empfehlungsdaten anzureichern und die Frequenz der Angebotsanzahl zu kontrollieren.

Die Klasse **Test-Klasse** dient zum Testen der eingebauten Logik. Sie generiert kontrollierte Test-Klick-Daten und Test-Lagerbestände. Im Kapitel Evaluation wird näher auf die Testfälle sowie ihre Auswertung eingegangen.

Zur Ausführung des Flink-Programms muss ein Replica-Set einer MongoDB bestehen und der Zookeeper Service, Apache Kafka, eine Schema-Registry sowie der Mongo-Kafka-Connector als auch die Ereignisquelle und ein Flink-Cluster gestartet werden. Die Versionen der verwendeten Programme befinden sich in der README.md des Java-Projektes.

### 5.2.2 Ereignisquellen

Die Ereignisquellen lassen sich in die Stammdaten und Empfehlungen, die in der MongoDB vorhanden sind, sowie in die Klick-Ereignisse und Bestandsveränderungen, die in der Java-Klasse Ereignisquelle generiert werden, einteilen. Dadurch dass bei der Recherche nach vorhandenen Daten zu Kundeninteraktionen keine adäquaten Beispieldatensätze gefunden werden konnten, wird neben der Bereitstellung der Daten kurz auf ihre Erstellung eingegangen.

**Daten in der MongoDB**
In der MongoDB sollen sich die Kunden-, Produkt- sowie Empfehlungsdaten befinden. 5.000 Kundendaten werden unter Inanspruchnahme der Website Migango.de [41] generiert und in Excel um eine Kunden-ID ergänzt. 3.000 Produktdaten werden in Excel zufällig aus einer Auswahl von Marken und Farben, Geschlecht, Preisspanne sowie einer Verkaufsorganisation (Standard, Sale, Neu) generiert. Die Verkaufsorganisation Standard

ist dabei am häufigsten vertreten. Die Produkte sind zudem den Kategorien Bekleidung, Schuhe und Accessoires sowie weiteren Unterkategorien zugeordnet.

Auf Grundlage der Empfehlungsregeln (siehe Tabelle 4.5) werden je Regel für alle relevanten Produkte drei zu den Eigenschaften passende Produkte zufällig ausgewählt und ihre IDs zusammengefügt. Da es drei Empfehlungsregeln gibt, entstehen entsprechend drei Tabellen zu Empfehlungen. Beispielsweise besteht die Tabelle zu E2 aus allen Produkten unter 50 Euro mit jeweils drei Produktempfehlungen, deren Preis ebenfalls unter 50 liegt und ungleich der Farben Schwarz, Grau oder Weiß sind sowie zum Geschlecht des Ursprungsproduktes passen.

Die Datentabellen werden in einer CSV-Datei gesichert und anschließend in die MongoDB geladen. Aus der MongoDB gelangen die Datensätze über Apache Kafka in die Ereignisverarbeitung. Dafür ist es notwendig, dass die MongoDB auf einem Replica-Set läuft. Veränderungen der Datenbasis werden automatisch erkannt und über das entsprechende Kafka Topic in Flink geladen.

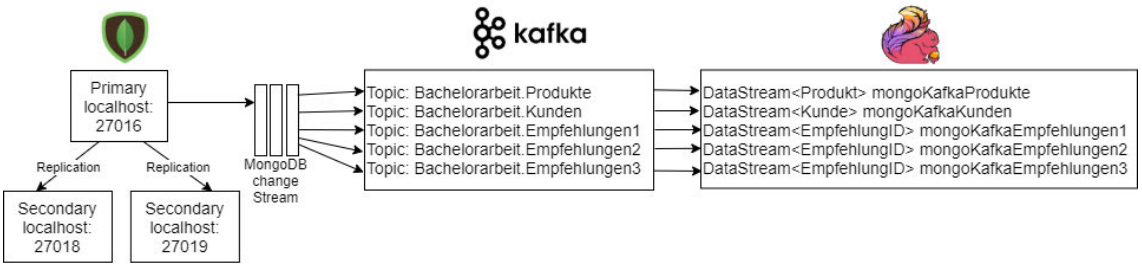



Abbildung 5.1: Apache-Kafka-Connector (eigene Darstellung)

Da Flink die eintreffenden Daten unmittelbar nach Erhalt verarbeitet und anreichert, ist es entscheidend, dass Flink bereits alle eintreffenden IDs bekannt sind. Dies bedeutet, Empfehlungen sollten Flink erst dann bereitgestellt werden, wenn die entsprechenden Produkte zuvor eingetroffen sind. Gleiches gilt für die im Folgenden thematisierten Kunden-Klicks und Bestandsveränderungen.

**Daten der Java-Klasse Ereignisquelle**

Damit Datenströme Flink erreichen, werden Klick-Ereignisse sowie Bestandveränderungen kontinuierlich von der im Java-Projekt vorhandenen Klasse Ereignisquelle generiert. Dabei werden unter Inanspruchnahme der Kunden- und Produktdaten der MongoDB die entsprechenden Ereignisse generiert und als POJOs der Klassen Klick-Datum und

Lagerbestand aufgebaut. Die Bestandsmenge des Lagerbestands entspricht einer zufälligen Menge zwischen 1 und 20. Der Klick-Typ des Klick-Datums wird zufällig aus den Klick-Typen Anschauen, Merken und Kaufen ermittelt. Die Wahrscheinlichkeiten des Auftretens liegen beim Typ Anschauen bei 70%, beim Typ Merken bei 20% und entsprechend beim Typ Kaufen bei 10%. Die generierten Objekte erreichen Flink über die Client-Server-Kommunikation mit Sockets.

Zur Vereinfachung wird davon ausgegangen, dass sich 1000 Kunden gleichzeitig für 15 Minuten auf der Website befinden. Im Durchschnitt klicken sie dabei drei Produkte pro Minute an. Somit erreichen Flink innerhalb von einer Sekunde 50 Klicks bzw. 3000 Klicks pro Minute. Bestandsveränderungen treffen dagegen mit einer geringeren Frequenz ein. Pro Sekunde erreichen ca. 10 Bestandsveränderungen Flink. Auf Grund der Tatsache, dass sich Kunden vermutlich Produkte mehrmals anschauen, wird die maximale Menge an verschiedenen Produkten auf 35 begrenzt. Dadurch wird zusätzlich die Wahrscheinlichkeit erhöht, dass die definierten Regeln zu den Angeboten und Empfehlungen erfüllt werden.

### 5.2.3 Ereignisverarbeitung

Im Mittelpunkt der Anwendung steht die Ereignisverarbeitung mit Apache Flink. Im Folgenden werden relevante Konzepte Flinks und Source Code zur Umsetzung der in Kapitel 4.4 definierten Logik nahegelegt.

#### 5.2.3.1 Programmaufbau von Flink

Beginnend wird kurz auf den allgemeinen Programmaufbau von Flink eingegangen. Zu Beginn jedes Flink Programms muss die Ausführungsumgebung eingerichtet werden, in der Operatoren ausgeführt werden können.

```
StreamExecutionEnvironment env = StreamExecutionEnvironment.
    getExecutionEnvironment();
```

Listing 5.1: Ausführungsumgebung

Zu den grundlegenden Komponenten von Flink zählen die Source, Transformation und Sink Operatoren. Ereignisse erreichen Flink als Datenstrom (Data Stream) über ein oder mehrere Source Operatoren und werden anschließend mittels Transformationsoperatoren

auf Ereignismuster überprüft. Jede durchgeführte Transformation erstellt dabei einen neuen Data Stream. Vollständig verarbeitete Datenströme erreichen schließlich die Sink Operatoren, welche die Verbindung zu anschließenden Anwendungssystemen stellen.

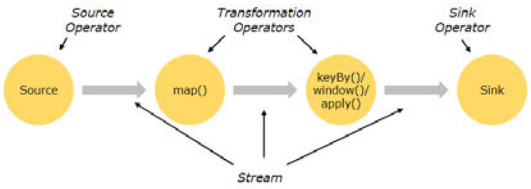



Abbildung 5.2: Apache Flink: Streaming Dataflow [9]

Mit dem folgenden Befehl am Ende des Flink Programms werden die definierten Operatoren schlussendlich ausgeführt.

```
env.execute("Flink Ereignisverarbeitung");
```

Listing 5.2: Ausführungsbefehl

#### 5.2.3.2 Zeiten

Apache Flink unterstützt verschiedene Zeiten bei der Verarbeitung der Datenströme. Zur Auswahl stehen die Processing-Time (Verarbeitungszeit), die Event-Time (Ereigniszeit), sowie die Ingestion-Time (Aufnahmezeit).

| | |
|---|---|
| **Processing-Time** | verwendet die Systemzeit der Maschine. |
| **Event-Time** | verwendet die Zeit des Auftretens des Ereignisses. Als Voraussetzung ist es zwingend notwendig, dass die eintreffenden Ereignisse mit einem Zeitstempel des Entstehungszeitpunktes versehen sind. |
| **Ingestion-Time** | verwendet die Zeit, zu der die Ereignisse Flink erreichen. Dafür werden die eintreffenden Ereignisse beim Empfang mit der aktuellen Zeit versehen. |

Tabelle 5.1: Zeiten in Apache Flink

Da einige Operatoren (z.B. der Interval Join verwendet in A5) bislang ausschließlich mit der Event-Time verwendet werden können und die Klick-Daten sowie Lagerbestände

ihren Entstehungszeitpunkt enthalten, wird in dem Prototypen die Event-Time berücksichtigt.

#### 5.2.3.3 Source Operatoren

**Source: Apache Kafka**
Bevor die Klick- und Bestands-Ströme Flink erreichen, müssen die Produkt-, Kunden- sowie Empfehlungsdaten eintreffen. Wie in Kapitel 5.2.2 erwähnt, sollen die Daten Flink über Apache Kafka erreichen. Hierfür ist es zunächst notwendig die Dependencies des Maven-Projektes um den Kafka Connector zu erweitern. Anschließend kann dieser in die Klasse Ereignisverarbeitung importiert und mit folgendem Befehl können die Daten aus den Kafka-Topics Flink zugänglich gemacht werden.

```
DataStream<Produkt> mongoKafkaProdukte = env.addSource(
        new FlinkKafkaConsumer("Bachelorarbeit.Produkte",
        new ProduktDeserializationSchema(), properties)
                .setStartFromEarliest());
```

Listing 5.3: Kafka Connection für das Produkte-Topic

Ein Flink-Kafka-Consumer enthält dabei den Namen des Kafka-Topics, ein Deserialization Schema, um die eintreffenden Kafka-Daten in POJOs umzuwandeln, sowie die Einstellung auf welchem Port Apache Kafka läuft. Der anschließende Befehl:
**.setStartFromEarliest())** sorgt dafür, dass beim Start der Anwendung alle in der Datenbank vorhandenen Daten in Flink geladen werden. Entsprechend der Kafka Connection für das Topic der Produkte werden ebenfalls die Kundendaten sowie die verschiedenen Empfehlungen in Flink geladen.

**Source: Klasse Ereignisquelle.java**
Sobald die Stammdaten und Empfehlungen Flink vollständig erreicht haben, werden die Klick-Daten sowie Lagerbestände von der Ereignisquelle generiert und erreichen Flink über die Sockets. Auch hierbei werden die eintreffenden Daten in die entsprechenden POJOs umgewandelt.

```
DataStream<KlickDatum> klickStream = env.socketTextStream("localhost", 9099)
        .map(new MapFunction<String, KlickDatum>() [...] );
```

Listing 5.4: Verbindung zum Klick-Socket

#### 5.2.3.4 Anreicherung der Klick-Daten

Eintreffende Klick-Daten werden über die Produkt-ID – auslesbar aus der Produkt-URL – und über die Kunden-ID mit den zur Verfügung stehenden Produkt- und Kundendaten angereichert. Der Operator Connect verbindet hierfür im ersten Schritt die beiden nach der Produkt-ID gruppierten Data-Streams der Klick-Daten sowie der Produktdaten und erzeugt aus diesen mit Hilfe der im Package States vorhandenen Klasse Connect-Klick-Produkt einen Data-Stream bestehend aus mit Produktinformationen angereicherten Klick-Daten.

```
DataStream<KlickProduktKunde> klickProduktKundeData = KlickStream
        .keyBy(c -> c.getProduktID())
        .connect(mongoKafkaProdukte.keyBy(produkt -> produkt.getProduktID())
        .process(new ConnectKlickProdukt())  [...] ;
```

Listing 5.5: Anreicherung der Klick-Daten

Im zweiten Schritt wird der neu erzeugte Data Stream erneut über den Connect Operator und mittels der Klasse Connect-Klick-Produkt-Kunde mit Kundendaten angereichert. Der neu entstandene Data-Stream enthält Ereignisse der Klasse Klick-Produkt-Kunde, die alle Klick-Informationen sowie nähere Informationen zu dem angeklickten Produkt und dem entsprechenden Kunden enthält.

Die Klassen Klick-Produkt und Klick-Produkt-Kunde sind äquivalent aufgebaut und erben von der Superklasse CoProcessFunktion, die aus den Elementen von zwei Data-Streams einen Data-Stream erzeugt.

Beide Klassen enthalten einen Value-State. States gehören bei Flink zu den zustandsbehafteten Operatoren, die es ermöglichen, Ereignisse über einen längeren Zeitraum zu speichern. Ein Value-State lässt sich den Keyed States zu ordnen und kann daher nur für gruppierte Data-Streams angewendet werden. In dem Value State Produkt-Data-State (bzw. Kunde-Data-State) befindet sich zum Schlüssel Produkt-ID (bzw. Kunden-ID) das gesamte Produkt-Objekt (bzw. Kunden-Objekt).

Während die Methode processElement1 für das eigentliche Verbinden der Datenströme zuständig ist, sorgt die Methode processElement2 dafür, dass sich im State immer die aktuellen Produkt- bzw. Kundendaten befinden. Dadurch kann gewährleistet werden, dass Veränderungen in der Datenbank bei der Verarbeitung in Flink berücksichtigt werden können.

```
public class ConnectKlickProduktKunde extends CoProcessFunction<KlickProdukt
    , Kunde, KlickProduktKunde> {
private ValueState<Kunde> kundeDataState = null; [...]

public void processElement1(KlickProdukt KlickProdukt, Context context,
    Collector<KlickProduktKunde> out) throws Exception {
        KlickDatum KlickDatum = KlickProdukt.getKlickDatum();
        Produkt produkt = KlickProdukt.getProdukt();
        out.collect(new KlickProdutKunde(KlickDatum, produkt, kundeDataState
            .value()));
}
public void processElement2(Kunde kunde, Context context, Collector<
    KlickProduktKunde> collector) throws Exception {
        kundeDataState.update(kunde); }}
```

Listing 5.6: Ausschnitt der Klasse Connect-Klick-Produkt-Kunde

**Event Time**

Neben der Anreicherung muss für die Event-Time die Zeit definiert und Watermarks ergänzt werden. Die Ereigniszeit wird dem Zeitstempel des Klick-Ereignisses entnommen. Watermarks werden verwendet, um das Fortschreiten der Ereigniszeit zu kontrollieren. Einem Operator, der Ereignisse stündlich verarbeitet, muss bspw. bekannt sein, wann eine Stunde abgelaufen ist. Vereinfacht wird davon ausgegangen, dass die Klick-Ereignisse und Lagerbestände nur in geordneter zeitlicher Reihenfolge eintreten.

```
.assignTimestampsAndWatermarks(new AscendingTimestampExtractor<
    KlickProduktKunde>(){
public long extractAscendingTimestamp(KlickProduktKunde value) {
        try {
        return value.getKlickDatum().getTimestamp().getTime();
        } [...] }}
```

Listing 5.7: Definition der Event-Time

#### 5.2.3.5 Angebote

Nachdem die Klick-Ereignisse aufbereitet wurden, werden sie bzgl. der definierten Angebote und Empfehlungs-Regeln überprüft.

**A1: Wenn ein Kunde den Newsletter noch nicht abonniert hat, dann weise ihn auf eine Preisreduzierung um 10% bei entsprechendem Abonnement hin.**

A1 wird wie folgt umgesetzt:
Der angereicherte Klick-Datenstrom wird im ersten Schritt mit Hilfe der Operation Filtern auf die Klick-Ereignisse begrenzt, die von einem Kunden ohne Newsletter-Abonnement getätigt wurden.

```
.filter(new FilterFunction<KlickProduktKunde>() {
@Override
public boolean filter(KlickProduktKunde value) throws Exception {
        return !value.getKunde().getNewsletter(); }}
```

Listing 5.8: Filter-Operator

Daraufhin wird der Datenstrom nach der Kunden-ID gruppiert und die Klasse Angebot-State aus dem Package States aufgerufen. Diese enthält vier Value-States, die sich auf die Angebote A1-A4 beziehen. In einem Value State ist ein Timestamp pro Kunden-ID hinterlegt, mit dem überprüft wird, ob der Kunde das entsprechende Angebot innerhalb der letzten drei Minuten bereits erhalten hat. Wenn dies nicht der Fall ist, erhält der Kunde das Angebot und der Timestamp des Kunden wird aktualisiert.

```
if (newsletterAngebot.value().before(new Timestamp(System.currentTimeMillis
    () - TimeUnit.MINUTES.toMillis(wartezeitNewsletter))) {
        newsletterAngebot.update(new Timestamp(System.currentTimeMillis()));
        out.collect(new Tuple4<Timestamp, Long, String, String>(new
            Timestamp(System.currentTimeMillis()),value.f1, "Angebot:", "
            Abonniere Newsletter und erhalte 10 Prozent Rabatt beim
            naechsten Einkauf!"));}
```

Listing 5.9: Zeitliche Überprüfung eines Angebots

Da es sich bei dieser Bachelorarbeit um die Entwicklung eines Prototypen handelt, wurde für die schnellere Überprüfung die Begrenzung der Angebotsfrequenz auf drei Minuten gelegt. In einer realen Anwendung würde die Frequenz niedriger sein.

**A2: Wenn ein Kunde als Neukunde gilt oder unter 25 Jahre alt ist und sich ein Produkt merkt, dann gewähre ihm einen kostenlosen Versand.**

A2 wird äquivalent zu A1 aufgebaut. Allerdings wird hierbei nach Kunden unter 25 und Neukunden sowie dem Klick-Typ Merken gefiltert. Das Alter des Kunden wird aus seinem Geburtsdatum und der Kunden-Status aus dem Registrierungsdatum ermittelt. Ein Kunde gilt als Neukunde, wenn er noch kein vollständiges Jahr registriert ist.

**A3: Wenn sich ein Kunde ein Produkt fünf Mal anschaut oder merkt, dann gewähre ihm einen Rabatt-Code von 10% auf das spezielle Produkt.**

Wie in Kapitel 2.2.1.6 Auswertungsfenster erläutert, bieten Fenster die Möglichkeit unendliche Datenströme in endlichen Mengen zu verarbeiten. Flink bietet vier verschiedene Window Assigner an, die für die Zuweisung der Ereignisse zu Fenstern zuständig sind.

**Tumbling Windows** ordnen die Ereignisse entsprechend der Zeit einem Fenster zu. Eine vorgegebene Fenstergröße bestimmt wie häufig ein Fenster ausgewertet und ein neues eröffnet wird.
**Sliding Windows** können sich im Gegensatz zu Tumbling Windows überlappen. Ereignisse können somit mehreren Fenstern angehören und dadurch mehrfach verarbeitet werden. Neben einer festen zeitlichen Fenstergröße wird ein weiterer Parameter verwendet, um die Überlappungszeit zwischen zwei Fenstern festzulegen.
**Session Windows** besitzen anders als Tumbling und Sliding Windows keine feste Start- und End-Zeit. Anstelle des Schließens des Fensters nach Erreichen einer definierten Größe schließen sich die Fenster, wenn für eine bestimmte Dauer keine neuen Ereignisse eingetroffen sind. Nachfolgende Ereignisse werden dann einem neuen Fenster zugewiesen.
**Global Windows** enthalten pro Gruppierung eines Datenstroms nur ein Fenster. Benutzerdefinierte Trigger werden verwendet, um Berechnungen durchzuführen.

Tabelle 5.2: Window Assigner in Apache Flink

Für die Umsetzung von A3 bietet sich die Verwendung von Global Windows an. Hierfür wird ein Trigger definiert, der alle fünf Ereignisse die Verarbeitung startet. Bevor jedoch der Window Assigner angewendet wird, wird der angereicherte Klick-Datenstrom auf die Klick-Typen Anschauen und Merken gefiltert und nach der Kunden-ID sowie der Produkt-ID gruppiert. Für jeden Kunden ergibt sich somit eine individuelle Anzahl an Fenstern auf Basis der von ihm betrachteten und gemerkten Produkte. Mit Hilfe der von Flink bereitgestellten ProcessingWindowFunction können alle durch den Trigger ausgelösten Ereignisse zugänglich gemacht und auf das Angebotsmuster überprüft werden. Im letzten Schritt folgt die Überprüfung, ob der Kunde in den letzten drei Minuten bereits ein A3 Angebot erhalten hat.

```
.countWindow(5).process(new ProcessWindowFunction<Tuple3<KlickProduktKunde,
    Long, Long>, Tuple3<KlickProduktKunde, Long, String>, Tuple,
    GlobalWindow>() {
public void process( [...] ) });
```

Listing 5.10: Global Window Assigner

**A4: Wenn ein Kunde sich innerhalb von zehn Klicks mind. sechs Produkte im Sale oder sich mind. sechs neue Produkte angeschaut oder gemerkt hat, dann sende ihm einen 10% Rabatt-Code auf die jeweilige Verkaufsorganisation.**

So wie A2 im Aufbau A1 ähnelt, gleicht die Struktur von A3 auch A4. Lediglich die ausschließliche Gruppierung nach der Kunden-ID sowie die Angebotslogik unterscheiden sich.

**A5: Wenn ein Kunde sich ein Produkt angeschaut oder gemerkt hat, dann weise ihn bei geringer Verfügbarkeit darauf hin. Unter einer geringen Verfügbarkeit ist ein Bestand unter fünf zu verstehen.**

A5 wird in zwei Varianten ausgeführt. Während der Kunde bei Variante eins einen Hinweis für ein soeben angeschautes oder gemerktes Produkt mit einer Bestandsmenge unter fünf erhält, schickt Variante zwei nach dem Erhalt einer Bestandsveränderung eines Produktes auf unter fünf allen Kunden, die innerhalb der letzten zwei Minuten das Produkt angeschaut oder gemerkt haben, einen Hinweis.

Für Variante eins wird genau wie bei der Anreicherung der Klick-Daten mit der Operation Connect und der Klasse Connect-Klick-Bestand das angereicherte Klick-Datum um die zum Produkt passende aktuelle Bestandsmenge erweitert. Ist der im Value-State gespeicherte Bestand des angeklickten oder gemerkten Produktes unter fünf, erhält der Kunde den Hinweis über eine geringe Bestandsmenge.

Bei Variante zwei werden der Datenstrom der angereicherten Klick-Daten und der Datenstrom der Bestandsveränderungen mittels eines Interval Joins verbunden. Ein Interval Join verbindet zwei Datenströme mittels eines Schlüssels und setzt die beiden Ströme bzgl. ihrer Timestamps in ein relatives Zeitintervall. In Bezug auf den Prototypen bilden die Produkt-IDs die Schlüssel und der Timestamp des Klick-Ereignisses gibt die Ausgangsbasis für die Betrachtung vor. Geprüft wird, ob die Bestandsmenge des angeklickten Produktes innerhalb der nächsten zwei Minuten unter fünf liegt.

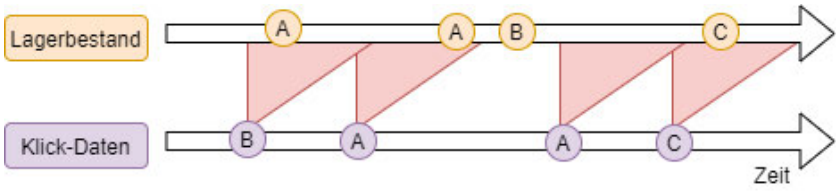



Abbildung 5.3: Interval Join mit den Beispiel-Produkten A, B und C (angelehnt an [10])

Im Gegensatz zur ersten Variante, die immer ausgeführt wird sobald ein Kunde ein Produkt mit einem Bestand unter fünf anklickt, wird bei Variante zwei im Anschluss überprüft, ob der Kunde für das entsprechende Produkt innerhalb der letzten drei Minuten bereits benachrichtigt wurde.

#### 5.2.3.6 Empfehlungen

Wie in Kapitel 5.2.3.3 beschrieben, erreichen die Empfehlungen Flink ebenfalls über den Kafka Connector. Jeder Empfehlungstyp hat dabei sein eigenes Kafka-Topic, die Flink einzeln als Datenströme erreichen (siehe Abbildung 5.1). Um die Empfehlungstypen auch bei der weiteren Verarbeitung auseinander halten zu können, werden die eintreffenden Ereignisse um eine Typ-Information ergänzt. Außerdem werden die einzelnen Produkt-IDs mit den vorhandenen Produkt-Informationen angereichert.

Die Empfehlungsregeln besitzen einen sehr ähnlichen Aufbau. E1 und E3 beginnen mit der Gruppierung des angereicherten Klick-Datenstroms nach der Kunden-ID. Im Anschluss folgt die Zuweisung der Ereignisse zu Fenstern durch den Windows Assigner Tumbling Window (siehe Tabelle 5.2) mit einer Fenstergröße von zwei Minuten. Da die Event Time in dem Prototypen verwendet wird, führt das Fenster die folgende Process-Window-Function erst aus, wenn ein Ereignis eintritt, dass einen Timestamp besitzt, der mindestens zwei Minuten älter ist als der Timestamp des Ereignisses, mit dem das Fenster eröffnet wurde. In der Process-Window-Function wird überprüft, ob die Logik der entsprechenden Empfehlungsregel zutrifft und entsprechend zum zuletzt relevanten Ereignis die passenden Produkt-Empfehlungen mit Hilfe von Connect und der Klasse Connect-Klick-Empfehlung angebunden.

```
.window(TumblingEventTimeWindows.of(Time.minutes(2))).process(new
    ProcessWindowFunction<KlickProduktKunde, KlickProduktKunde, Long,
    TimeWindow>() {
@Override
public void process(Long key, Context context, Iterable<KlickProduktKunde>
    elements, Collector<KlickProduktKunde> out) throws Exception { [...] }
    });
```

Listing 5.11: Tumbling Window Assigner

Während sich bei E1 und E3 nur die Logik in der Process Window Function unterscheidet, kommt bei E2 hinzu, dass der Klick-Datenstrom vor der Gruppierung nach der Kunden-ID auf Klicks von Kunden unter 30 gefiltert wird.

### 5.2.4 Ereignisbehandlung

Die vollständig verarbeiteten Angebote und Empfehlungen werden als **Tuple4<Timestamp, Long, String, String>** im Terminal ausgegeben. Der Zeitstempel gibt an, zu welchem Zeitpunkt das Angebot bzw. die Empfehlung von Apache Flink bereitgestellt wurde. Der Datentyp Long beinhaltet die Kunden-ID, darauf folgt die Information, ob es sich um eine Empfehlung oder ein Angebot handelt, und die letzte Zeichenkette beinhaltet die eigentliche Mitteilung an den Kunden.

# 6 Evaluation

Abschließend erfolgt die Bewertung des in Kapitel 4 konzeptionierten und in Kapitel 5 entwickelten Systems. Zuerst werden die fachlichen Angebots- sowie Empfehlungs-Regeln, definiert in Kapitel 4.4, getestet und anschließend die in Kapitel 3.3 definierten Anforderungen an eine Echtzeitanalyse im Rahmen einer E-Commerce Plattform überprüft. Zum Abschluss soll die konkrete Umsetzung des Systems auf seine Realitätsnähe untersucht werden.

## 6.1 Auswertung der Ergebnisse

Um das System auf die korrekte Ausführung der definierten Regeln testen zu können, werden mit Hilfe der Java-Klasse Testklasse kontrollierte Klick-Daten erzeugt. Die sich im Anhang befindende Tabelle (siehe A.2) verdeutlicht die durchdachten Testfälle und die erwarteten Ergebnisse. Für alle fünf Angebotsregeln sowie die drei Empfehlungsregeln werden Positiv- sowie Negativtests durchgeführt. Die Kontrolle der Ergebnisse erfolgt über die einzelnen Test-Kunden.

Beispielsweise wird A1 auf drei Use-Cases überprüft:

- Kunde1 hat Newsletter abonniert und erhält daher keine Mitteilung.
- Kunde2 hat keinen Newsletter abonniert und erhält daher eine Mitteilung.
- Kunde2 hat innerhalb der letzten drei Minuten bereits die Aufforderung den Newsletter zu abonnieren erhalten und erhält daher nicht erneut eine Mitteilung.

Die nach der Ausführung der Testklasse im Terminal ausgegebenen Kundenmitteilungen sowie deren Klick-Verhalten und Bestandsveränderungen wurden mit Hilfe von Excel nach den zu den Regeln gehörenden Kunden gefiltert und in einzelne Tabelle aufgesplittet (siehe A.1 Test_Log.xlsx). Für einen Testfall irrelevante Mitteilungen auf Grund von anderer Regeln wurden zur Übersichtlichkeit ausgegraut und erwartete Mitteilungen grün

hinterlegt. Alle positiven und negativen Testfälle können ihren Vorgaben entsprechend erfüllt werden und zeigen somit die fachlich korrekte Ausführung der definierten Logik.

| Erfüllt | | |
|---|---|---|
| | | **Angebote und Hinweise** |
| ✓ | **A1** | Wenn ein Kunde den Newsletter noch nicht abonniert hat, dann weise ihn auf eine Preisreduzierung um 10% bei entsprechendem Abonnement hin. |
| ✓ | **A2** | Wenn ein Kunde als Neukunde gilt oder unter 25 Jahre alt ist und sich ein Produkt merkt, dann gewähre ihm einen kostenlosen Versand. |
| ✓ | **A3** | Wenn sich ein Kunde ein Produkt fünf Mal anschaut oder merkt, dann gewähre ihm einen Rabatt-Code von 10% auf das spezielle Produkt. |
| ✓ | **A4** | Wenn ein Kunde sich innerhalb von zehn Klicks mind. sechs Produkte im Sale oder sich mind. sechs neue Produkte angeschaut oder gemerkt hat, dann sende ihm einen 10% Rabatt-Code auf die jeweilige Verkaufsorganisation. |
| ✓ | **A5** | Wenn ein Kunde sich ein Produkt angeschaut oder gemerkt hat, dann weise ihn bei geringer Verfügbarkeit darauf hin. Unter einer geringen Verfügbarkeit ist ein Bestand unter fünf zu verstehen. |
| | | **Empfehlungen** |
| ✓ | **E1** | Wenn ein Kunde innerhalb von zwei Minuten mind. zu 50% Produkte der gleichen Kategorie angeklickt hat, dann empfehle ihm drei Produkte aus der Kategorie, die zum Geschlecht des zuletzt betrachteten Produktes der Kategorie passen. |
| ✓ | **E2** | Wenn ein Kunde unter 30 Jahre alt ist und innerhalb von zwei Minuten zu mind. 50% Produkte unter 50€ angeklickt hat, dann empfehle ihm drei farbenfrohe Produkte unter 50€, die zum Geschlecht des zuletzt betrachteten Produktes unter 50€ passen. |
| ✓ | **E3** | Wenn ein Kunde innerhalb von zwei Minuten zu mind. 70% Produkte über oder gleich 50€ angeklickt hat, dann empfehle ihm drei Produkte über 50€, die sich nicht im Sale befinden und zum Geschlecht des zuletzt betrachteten Produktes über oder gleich 50€ passen. |

Tabelle 6.1: Erfüllte Angebote und Empfehlungen

## 6.2 Anforderungsabgleich

Im Folgenden soll kontrolliert werden, ob die aufgestellten Anforderungen in Kapitel 3.3 eingehalten werden.

**Funktionale Anforderungen**
Die unter Kapitel 6.1 erfüllten Angebots- und Empfehlungsregeln zeigen, dass die eintreffenden Klick-Ereignisse korrekt auf Verhaltensmuster durchsucht werden und die Verarbeitung kundenindividuell funktioniert. Auf die Effizienz soll bei der Überprüfung der nicht funktionalen Anforderungen eingegangen werden.

Zuzüglich zu den Datenströmen der Klick-Daten und der Lagerbestandsveränderungen erreichen auch die Veränderungen in der MongoDB Apache Flink, sodass bei der Verarbeitung der Ereignisse gespeicherte Kunden- und Produktdaten Berücksichtigung finden können. Die Anbindung durch den Kafka-Connector ermöglicht eine stets aktuelle Stammdatenbasis. Allerdings wird zur Vereinfachung bei Veränderungen in der Datenbank ausschließlich das veränderte Dokument ohne weitere Informationen – wie z.B. ein Zeitstempel des Veränderungszeitpunktes - an Flink versendet. Daraus ergibt sich, dass nicht sichergestellt werden kann, ob alle Klick-Ereignisse mit den korrekten Stammdaten angereichert werden. Beispielsweise könnten verspätete Klick-Ereignisse mit zu aktuellen Stammdaten angereichert werden.

Auf Grund dessen, dass die Empfehlungsregeln zu der Datenstruktur der Produkte passen, können vereinfacht in Abhängigkeit zu der jeweiligen Regel ähnliche Produkte zusammengestellt werden. Die lose Kopplung zwischen den einzelnen Angebots- und Empfehlungsregeln führt dazu, dass Regeln unabhängig voneinander modifiziert werden können und die Logik um weitere Regeln ergänzt werden kann. Somit wird auch FA6 als erfüllt betrachtet. Während eine klassische Kunden-Clusterung in der Arbeit nicht durchgeführt wird, zeigen Regeln die bspw. nur für Kunden im Alter von unter 25 gelten, dass Kundengruppen dennoch unterschiedliche Angebote erhalten können. Wie ebenfalls die Testergebnisse aus Kapitel 6.1 widerspiegeln, gibt es eine zustandsbehaftete Operation, die dafür sorgt, dass die einzelnen Angebotstypen nicht durchweg den Kunden erreichen, sondern in einer fest gelegten Zeitspanne nur einmal auftreten. Die Anzahl an unterschiedlichen Angebotstypen wird jedoch nicht begrenzt.

Die Bereitstellung der Angebote und Empfehlungen an den Kunden erfolgt musterhaft für anschließende Systeme über das Terminal. Im Gegensatz zu externen Datenströmen, die in dem Prototypen vernachlässigt werden, erreicht Flink beispielhaft ein weiterer

interner Datenstrom. Die eintreffenden Bestandsveränderungen lassen sich mit den Klick-Ereignissen verbinden und können somit bei der Verarbeitung berücksichtigt werden.

| Erfüllt | | Funktionale Anforderungen |
|---|---|---|
| ✓ | **FA1** | Kundeninteraktionen müssen in Echtzeit nach Verhaltensmustern untersucht werden. |
| ✓ | **FA2** | Kunden müssen identifiziert und ihre Interaktionen unabhängig voneinander verarbeitet werden. |
| ✓ | **FA3** | Detaillierte und aktuelle Kundendaten stehen dem System zur Verfügung. |
| ✓ | **FA4** | Standardisierte und aktuelle Produktdaten stehen dem System zur Verfügung. |
| ✓ | **FA5** | Auf Basis der Produktdaten lassen sich Empfehlungen ableiten. |
| ✓ | **FA6** | Die Angebots- und Empfehlungslogik lässt sich modifizieren und erweitern. |
| (✓) | **FA7** | Kunden können anhand ihrer Eigenschaften und ihres Verhaltens segmentiert werden und entsprechend passende Angebote erhalten. |
| (✓) | **FA8** | Ein Kunde erhält eine begrenzte Anzahl an Angeboten pro Session. |
| (✓) | **FA9** | Empfehlungen und Angebote werden Kunden bereitgestellt. |
| | **FA10** | Ein Datenstrom externer Datenquellen kann integriert und verarbeitet werden. |
| ✓ | **FA11** | Ein Datenstrom interner Datenquellen kann integriert und verarbeitet werden. |

Tabelle 6.2: Erfüllte Funktionale Anforderungen

**Nicht-Funktionale Anforderungen**

Nachdem die funktionalen Anforderungen begutachtet wurden, folgt die Revision der nicht funktionalen Anforderungen. Die Anforderungen zur Zuverlässigkeit sind wie in Kapitel 4.6.4 beschrieben umsetzbar, finden in dem Prototypen jedoch keine Berücksichtigung.

Die Anforderungen zur Effizienz können hingegen nicht vernachlässigt werden. Sie bilden den essentiellen Mehrwert eines Echtzeitsystems. Um die Effizienz zu untersuchen, wird die Klasse Ereignisquelle für ca. 20 Minuten ausgeführt. Innerhalb dieser Zeit verarbeitet das System 58.733 Klick-Daten und 11.747 Bestandsveränderungen. Ausgegeben werden 15.085 Empfehlungen sowie 65.422 Angebote und Hinweise. Die ungewöhnlich

große Anzahl an Angeboten erscheint in der Realität als unrealistisch, zeigt allerdings hier die effiziente Verarbeitung von großen Datenmengen. Allein zu A5 zählen 55.635 der ausgegebenen Angebote und Hinweise. Diese unerwartet hohe Anzahl ergibt sich daraus, dass die Simulation pro 15 Minuten ausschließlich 35 verschiedene Produkte auswählt, die angeklickt werden können und deren Bestandsmenge sich verändert. Darüber hinaus verändert sich der Bestand bei etwa zehn Produkten pro Sekunde. Der Bestand der Produkte befindet sich zu jeder Zeit zwischen eins und 20.

Wie in NFA5 gefordert simuliert die Ereignisquelle 1000 sich gleichzeitig auf der Website befindende Kunden, die im Durchschnitt drei Klicks pro Minute tätigen. Die Auswertung der Ergebnisse der ausgeführten Ereignisquelle verdeutlicht, dass sowohl zu Beginn als auch nach einer Laufzeit von mehreren Minuten Angebote und Empfehlungen korrekt ausgegeben werden. Die Verarbeitungszeit je Angebot oder Empfehlung beträgt durchgehend weniger als 300 Millisekunden, i.d.R. liegt sie unterhalb von 200 Millisekunden. Die Verarbeitungszeit wird anhand des Zeitraums zwischen dem Erzeugungszeitpunkt des zuletzt relevanten Ereignisses und dem Zeitpunkt der ausgegebenen Mitteilung gemessen.

Genau wie die einzelnen Angebots- und Empfehlungsregeln lassen sich die Komponenten des Systems unabhängig voneinander modifizieren sowie um weitere Komponenten erweitern. Die Kommunikation verläuft stets im Push-Modus (thematisiert in Kapitel 2.2.3.1) und unterstreicht somit u.a. die lose Kopplung der einzelnen Komponenten. Aus dem vorherigen Kapitel 6.1 wird außerdem deutlich, dass NFA8 ebenfalls erfüllt wird. Tests in Bezug auf die genutzten Ressourcen wurden vernachlässigt. Es lässt sich lediglich festhalten, dass das System über einige Minuten hinweg unproblematisch mit den vorhandenen Ressourcen (siehe Kapitel 4.6.4) lauffähig bleibt.

Mit einem kritischen Blick auf das Kapitel Umsetzung können allerdings einige Punkte aufgezählt werden, bei denen sich die Belastung der Arbeitsspeicherkapazität optimieren lässt. Beispielsweise erreichen Flink keine gelöschten Dokumente der MongoDB. Diese bleiben überflüssig in den States für die Anreicherung der Klick-Daten gespeichert. Ein weiterer Aspekt bezieht sich auf die Global Windows (genutzt in A3 und A4), die niemals geschlossen werden.

| Erfüllt | | Nicht-Funktionale Anforderungen |
|---|---|---|
| | **NFA1** | Der Server, auf dem der Webshop gehostet wird, garantiert eine Verfügbarkeit von mindestens 99,99%. |
| | **NFA2** | Das System verfügt über keine Single Points of Failure. |
| | **NFA3** | Es existiert ein Wiederherstellungsmechanismus, der nach einem Ausfall das System erneut aufbauen kann. |
| ✓ | **NFA4** | Innerhalb von maximal drei Sekunden können passende Empfehlungen und gezielte Angebote analysiert und dem Kunden bereitgestellt werden. |
| ✓ | **NFA5** | Zur gleichen Zeit können Daten von 1000 Kunden, die im Durchschnitt drei Klicks pro Minute tätigen, verarbeitet werden. |
| ✓ | **NFA6** | Ein kontinuierlich eintreffender Datenstrom kann mit den vorhandenen Ressourcen verarbeitet werden. |
| ✓ | **NFA7** | Die einzelnen Komponenten der Anwendung sind unabhängig voneinander modifizier- und erweiterbar. |
| ✓ | **NFA8** | Das System lässt sich bezüglich seiner Funktionalitäten testen. |
| | **NFA9** | Das System lässt sich hinsichtlich der genutzten Ressourcen testen. |

Tabelle 6.3: Erfüllte Nicht-Funktionale Anforderungen

## 6.3 Kritische Betrachtung

Abschließend soll hier eine kurze Gesamtbewertung des entstandenen Systems vorgenommen werden. Hierfür ist es sinnvoll sich an den Titel der Thesis zu erinnern: Personalisierung im E-Commerce unter Verwendung von Realtime Streaming Analytics. Demzufolge wurde erfolgreich eine Echtzeitanalyse für den E-Commerce Bereich entwickelt, die unter der Berücksichtigung von eintreffenden Datenströmen und gespeicherten Stammdaten personalisierte Angebote und Empfehlungen Kunden in Echtzeit bereitstellt. Wie in Kapitel 6.2 aufgezeigt, wird das Gros der an das System gestellten Anforderungen umgesetzt. Auch Apache Flink erweist sich durch seine effiziente Verarbeitung großer Datenmengen, seiner Unterstützung der losen Kopplung der Fachlogik sowie durch seine korrekte Umsetzung aller aufgestellten Regeln (siehe Kapitel 6.1) als geeignetes Echtzeit-Framework.

Trotz des allgemein gelungenen Prototypen lassen sich einige Aspekte nennen, die in einem in der Praxis verwendeten System nicht vernachlässigt werden dürfen oder hin-

terfragt werden müssten. So stellen die einbezogenen Datenströme nur eine stark vereinfachte Sicht auf mögliche eintreffende Ereignisse dar. Weder werden die große Menge an unterschiedlichen Klick-Möglichkeiten eines Webshops berücksichtigt noch weitere für die Angebotsauswahl interessante Attribute wie die Verweildauer auf einzelnen Seiten. Darüber hinaus treten in der Realität weitere interne sowie externe Ereignisse auf, die bei der Verarbeitung nicht außer Acht gelassen werden sollten (siehe Kapitel 3.1.1).

Um gezieltere Angebote sowie Empfehlungen stellen zu können, sollten zusätzlich zu den aktuellen Kundendaten auch ältere Session-Informationen wie bereits gekaufte oder gemerkte Produkte in die Analyse einbezogen werden. Die in Kapitel 2.1.5 angesprochenen psychografischen Kriterien eines Kunden werden ebenfalls vernachlässigt. Außerdem muss in der Praxis der Umgang mit nicht registrierten Kunden definiert sein. Empfehlungen ergeben sich im System ausschließlich aus zufällig kombinierten Produkten mit ähnlichen Eigenschaften. In der Realität sollten erprobte Empfehlungssysteme, wie in Kapitel 2.1.4 beschrieben, eingebunden werden.

Die in der Umsetzung konkret in Zahlen festgelegten Annahmen zur Anzahl der sich gleichzeitig auf dem Webshop befindenden Kunden sowie deren Verweildauer und Anzahl an Klicks pro Minute sind standardisiert (siehe Kapitel 5.2.2). In einem realen Umfeld ändern sich diese durchgehend und sind im Allgemeinen auch von der Ware und der Größe des Online Shops abhängig. Die eintreffenden Produktmengen ergeben auf Grund ihrer zufällig simulierten Bestände keine realistischen Sequenzen. So ist es bspw. möglich, dass der Bestand eines Produktes von 15 auf 5 fällt. Des Weiteren stellt die Implementierung der Dauer von drei Minuten zwischen zwei Angeboten eines Typen keine reelle Zeitspanne dar. Sie wurde gewählt, um das System in angemessener Zeitdauer überprüfen zu können.

Zuletzt soll die im Prototyp vernachlässigte Zuverlässigkeit thematisiert werden. Alle gestellten nicht funktionalen Anforderungen bzgl. dieser müssen in einem in der Praxis eingesetzten System Berücksichtigung finden. Darüber hinaus muss auch eine unerwartet oder selten hohe Auslastung, die z.B. durch speziell geschaltete Werbung oder durch besonders beliebte Shopping-Tage entstehen kann, getestet werden.

# 7 Fazit und Ausblick

Dieses Kapitel fasst die zentralen Erkenntnisse der Thesis zusammen und gibt einen kurzen Ausblick auf weiterführende Forschungsgebiete.

## 7.1 Fazit

Gerade in Zeiten der Corona Pandemie wird der Verkauf von Ware über das Internet noch präsenter, aber auch vorherige Tendenzen zeigen deutliche Zunahmen der Nutzerzahlen im E-Commerce weltweit. Die hohe Anzahl an potentiellen Kunden sowie der wachsende Konkurrenzkampf zwingen die Unternehmen zu leistungsstarken Systemen und einen im Mittelpunkt stehenden Kunden. Da das Angebot für einen Kunden groß ist, bleibt nur eine kurze Zeit, um diesen vom eigenen Webshop zu überzeugen. Mit personalisierten Maßnahmen können Kaufs- und Wiederbesuchsabsichten gesteigert werden. Unternehmen müssen sich zwangsläufig mit dem Thema Big Data auseinandersetzen, um aus der außerordentlich großen Anzahl an Kundeninteraktionen sowie weiteren eintreffenden Datenströmen einen Nutzen erzielen zu können.

Unter der Berücksichtigung von EDA mit CEP konnte mit Hilfe des Frameworks Apache Flink gezeigt werden, dass eine große Anzahl von Kundeninteraktionen verarbeitet werden kann und individuell abhängig von verhaltensorientierten und soziodemografischen Merkmalen Angebote und Empfehlungen ausgegeben werden können. Die lose Kopplung zwischen den Komponenten und einzelnen Regeln unterstützt eine schnelle und unkomplizierte Anpassung an die agile Umwelt. Eine korrekte und effiziente Verarbeitung des Prototypen konnte nachgewiesen sowie an das System gestellte Anforderungen weitgehend abgedeckt oder konzeptioniert werden. Somit konnte im kleinen Rahmen gezeigt werden wie ein Echtzeitsystem für eine E-Commerce Plattform zur Umsetzung von personalisierten Maßnahmen aussehen könnte.

Neben der erfolgreichen Entwicklung des Prototypen sollte jedoch auch mit einem Blick zurück auf die Kapitel 3 Analyse und 4 Konzeption die Komplexität sowie der große Umfang eines in der Realität verwendeten Systems nicht unterschätzt werden. Während der Prototyp nur einzelne Kundeninteraktionen berücksichtigt und zusätzlich ausschließlich die Lagerbestandsveränderungen als weiteren internen Strom den Prototyp passieren, müssen in der Praxis eine Vielzahl an weiteren Einflüssen Beachtung finden. Darüber hinaus sollten die empfohlenen Angebote und Empfehlungen auf analytischen Systemen wie z.B. Empfehlungssystemen aufbauen. Als letzter Punkt ist die hohe Arbeitsspeicherauslastung zu nennen, die auf Grund der Verarbeitung im Hauptspeicher bei Echtzeitsystemen nicht zu vernachlässigen ist.

## 7.2 Ausblick

Mit dieser Thesis wurde die Grundidee verfolgt ein Echtzeitsystem zu entwickeln, das personalisierte Angebote und Empfehlungen für eine E-Commerce Plattform bereitstellt. Dem Umfang geschuldet konnte sich ausschließlich mit einem stark vereinfachten System beschäftigt werden. Weiterführende Arbeiten könnten an diesem Standpunkt ansetzen und sich mit Empfehlungssystemen oder Kunden-Clusterungen befassen. Des Weiteren besteht die Möglichkeit ein vergleichbares System mit alternativen Streaming-Analytics Tools zu entwickeln und diese mit dem entwickelten Prototyp zu vergleichen. Ebenfalls könnten die nicht funktionalen Anforderungen - insbesondere die Zuverlässigkeit - stärker fokussiert werden.

Ferner existieren auf dem Markt weitere Methoden zur Personalisierung, die in dieser Arbeit vernachlässigt oder einzig in den Grundlagen und der Analyse erwähnt wurden. So können bspw. auch personalisierte Seiten-Layouts oder personalisierte Preise untersucht werden. Weitere interessante Aspekte, die in der Thesis nicht behandelt wurden, sind automatisierte Personalisierung mittels künstlicher Intelligenz oder der Einsatz von Augmented Reality im E-Commerce.

# Literaturverzeichnis


[1] About You: *Fashionshop AboutYou.* – URL https://www.aboutyou.de/. – Zugriffsdatum: 2021-02-01

[2] Aichele, C. ; Schönberger, M.: *E-Business - Eine Übersicht für erfolgreiches B2B und B2C.* Springer Vieweg, 2016. – ISBN 978-3-658-13686-4

[3] Alkatheri, S. ; Abbas, S. ; Siddiqui, M.: A Comparative Study of Big Data Frameworks. In: *International Journal of Computer Science and Information Security (IJCSIS)* (2019)

[4] Altrichter, J.: *Big Data, Smart Data – Lost Data?.* – URL https://blog.commerzbank.de/digitalisierung/18q2/studie-up-big-data.html. – Zugriffsdatum: 2020-11-26

[5] Amazon: *Produktdetailseite auf Amazon.de.* – URL https://www.amazon.de/ZTE-Smartphone-interner-Triple-Kamera-AI-Technologie/dp/B07W97H8ND/. – Zugriffsdatum: 2021-02-01

[6] Amazon: *Webshop Amazon.de.* – URL https://www.amazon.de/. – Zugriffsdatum: 2021-02-01

[7] Amazon Web Services: *Amazon Web Services.* – URL https://aws.amazon.com/de/. – Zugriffsdatum: 2021-03-21

[8] Anne, S.: *Web analytics with IBM Db2 EventStore: Ingest streaming web events data.* – URL https://www.ibmbigdatahub.com/blog/web-analytics-ibm-eventstore-ingest-streaming-web-events-data. – Zugriffsdatum: 2021-02-01

[9] Apache Flink: *Apache Flink – offizielle Website.* – URL https://flink.apache.org/. – Zugriffsdatum: 2021-02-01

[10] Apache Flink: *Joining Apache Flink.* – URL https://ci.apache.org/projects/flink/flink-docs-stable/dev/stream/operators/joining.html. – Zugriffsdatum: 2021-04-08

[11] Apache Kafka: *Apache Kafka – offizielle Website.* – URL https://kafka.apache.org/. – Zugriffsdatum: 2021-02-01

[12] Apache Spark: *Structured Streaming Programming Guide.* – URL https://spark.apache.org/docs/latest/structured-streaming-programming-guide.html#programming-model. – Zugriffsdatum: 2021-02-01

[13] Apache Storm: *Apache Storm – offizielle Website.* – URL https://storm.apache.org/. – Zugriffsdatum: 2021-02-01

[14] Arz, S.: *Persönlichkeitsbasierte Personalisierung im Mobile Commerce - Eine verhaltenswissenschaftliche Analyse am Beispiel von Supermarkt-Apps.* Springer Gabler, 2020. – ISBN 978-3-658-31818-5

[15] Backhaus, K. ; Erichson, B. ; Plinke, W. ; Weiber, R.: *Multivariate Analysemethoden - Eine anwendungsorientierte Einführung.* 15. Auflage. Springer Gabler, 2018. – ISBN 978-3-662-56654-1

[16] Balzert, H.: *Lehrbuch der Softwaretechnik - Entwurf, Implementierung, Installation und Betrieb.* 3. Auflage. Spektrum, 2011. – ISBN 978-3-8274-1706-0

[17] Beasley, M.: *Practical Web Analytics for User Experience.* Elsevier Inc., 2013. – ISBN 978-0-12-404619-1

[18] Bernhard, M. ; Mühling, T.: *Verantwortungsvolle KI im E-Commerce - Eine kurze Einführung in Verfahren der Künstlichen Intelligenz in der Webshop-Personalisierung.* Springer Gabler, 2020. – ISBN 978-3-658-29036-8

[19] Berns, K. ; Köpper, A. ; Schürmann, B.: *Technische Grundlagen Eingebetteter Systeme - Elektronik, Systemtheorie, Komponenten und Analyse.* Springer Vieweg, 2019. – ISBN 978-3-658-26515-1

[20] Betz, S. ; Müller, T. ; Wiese, H ; Siebers, B. ; Fischer, J. ; Zacek, M.: Personalisierung in der Praxis. In: *Praxis der Personalisierung im Handel.* Springer Gabler, 2017. – ISBN 978-3-658-16243-6

[21] Bruns, R. ; Dunkel, J.: *Event-Driven Architecture - Softwarearchitektur für ereignisgesteuerte Geschäftsprozesse*. Springer, 2010. – ISBN 978-3-642-02438-2

[22] Bruns, R. ; Dunkel, J.: *Complex Event Processing - Komplexe Analyse von massiven Datenströmen mit CEP*. Springer Vieweg, 2015. – ISBN 978-3-658-09898-8

[23] Chaffey, D.: *Digital Business and E-Commerce Management – Strategy, Implementation and Practice*. 6. Auflage. Pearson HigherEducation, 2014. – ISBN 978-0-273-78657-3

[24] Cloudera: *Apache Storm by Cloudera*. – URL https://de.cloudera.com/products/open-source/apache-hadoop/apache-storm.html. – Zugriffsdatum: 2021-02-01

[25] Databricks: *Apache Spark by Databricks*. – URL https://databricks.com/de/spark/about. – Zugriffsdatum: 2021-02-01

[26] Deges, F.: *Grundlagen des E-Commerce - Strategien, Modelle, Instrumente*. Springer Gabler, 2020. – ISBN 978-3-658-26319-5

[27] Engelhardt, J. ; Magerhans, A.: *eCommerce klipp & klar*. Springer Gabler, 2019. – ISBN 978-3-658-26503-8

[28] Fabisch, N.: Ethische Grenzen der Datennutzung im Marketing. In: *Data-driven Marketing - Insights aus Wissenschaft und Praxis*. Springer Gabler, 2020. – ISBN 978-3-658-29994-1

[29] Fasel, D. ; Meier, A.: *Big Data - Grundlagen, Systeme und Nutzungspotenziale*. Springer Vieweg, 2016. – ISBN 978-3-658-11588-3

[30] Gartner: *Big Data*. – URL https://www.gartner.com/en/information-technology/glossary/big-data. – Zugriffsdatum: 2020-11-26

[31] Glaser, c.: *Wettbewerbsfaktor Vertrieb bei Finanzdienstleistern - Ein ganzheitliches Konzept zur Sales Excellence*. 2. Auflage. Springer Gabler, 2017. – ISBN 978-3-658-15645-9

[32] Grosse Holtforth, D.: *Schlüsselfaktoren im E-Commerce - Innovationen, Skaleneffekte, Daten und Kundenzentrierung*. Springer Gabler, 2017. – ISBN 978-3-658-16433-1

[33] Hedtstück, U.: *Complex Event Processing - Verarbeitung von Ereignismustern in Datenströmen*. 2. Auflage. Springer Vieweg, 2020. – ISBN 978-3-662-61575-1

[34] Heinemann, G.: *Der neue Online-Handel - Geschäftsmodelle, Geschäftssysteme und Benchmarks im E-Commerce*. 11. Auflage. Springer Gabler, 2020. – ISBN 978-3-658-28203-5

[35] Holland, H.: Big Data - Chancen und Herausforderungen. In: *Digitales Dialogmarketing*. Springer Gabler, 2020. – ISBN 978-3-658-28973-7

[36] Höhfeld, S. ; Kwiatkowski, M.: Empfehlungssysteme aus informationswissenschaftlicher Sicht – State of the Art. In: *Information Wissenschaft & Praxis 5/2007* (2007)

[37] Institut LINK: Abschlussbericht zum Projekt: Repräsentative Verbraucherbefragung in der Gruppe der Internetnutzer / Sachverständigenrat für Verbraucherfragen beim Bundesministerium der Justiz und für Verbraucherschutz. 2016. – Forschungsbericht

[38] Kleuker, S.: *Grundkurs Datenbankentwicklung - Von der Anforderungsanalyse zur komplexen Datenbankanfrage*. 4. Auflage. Springer Vieweg, 2016. – ISBN 978-3-658-12337-6

[39] Lopez, M. ; Lobato, A. ; Duarte, C.: A Performance Comparison of Open-Source Stream Processing Platforms. In: *IEEE* (2016)

[40] Meier, A.: *Werkzeuge der digitalen Wirtschaft: Big Data, NoSQL & Co. - Eine Einführung in relationale und nicht-relationale Datenbanken*. Springer Vieweg, 2018. – ISBN 978-3-658-20336-8

[41] Migano: *Testdaten Generator*. – URL https://migano.de/testdaten.php. – Zugriffsdatum: 2021-04-06

[42] MongoDB: *MongoDB Connector for Apache Kafka*. – URL https://www.mongodb.com/kafka-connector. – Zugriffsdatum: 2021-02-01

[43] MongoDB: *MongoDB – offizielle Website*. – URL https://www.mongodb.com/de. – Zugriffsdatum: 2021-02-01

[44] Ng, A. ; Soo, K.: *Data Science – was ist das eigentlich?! - Algorithmen des maschinellen Lernens verständlich erklärt*. Springer, 2018. – ISBN 978-3-662-56775-3

[45] Otto: *Produktdetailseite auf Otto.de*. – URL https://www.otto.de/p/levis-v-shirt-perfect-tee-mit-kleinem-batwing-logo-813906757/#variationId=1061821343. – Zugriffsdatum: 2021-02-01

[46] Otto: *Webshop Otto.de.* – URL https://www.otto.de/. – Zugriffsdatum: 2021-02-01

[47] Reiknecht, J. ; Papp, S.: *BIG DATA in der Praxis - Lösungen mit Hadoop, Spark, HBase und Hive Daten speichern, aufbereiten, visualisieren.* 2. Auflage. Hanser, 2018. – ISBN 978-3-446-45396-8

[48] Schleusener, M.: Personalisierte Preise im Handel – Chancen und Herausforderungen. In: *Praxis der Personalisierung im Handel.* Springer Gabler, 2017. – ISBN 978-3-658-16243-6

[49] Schlüter, O. ; Will, A.: Personalisierung über alle digitalen Touchpoints – Vorgehensmodell am Beispiel otto.de. In: *Praxis der Personalisierung im Handel.* Springer Gabler, 2017. – ISBN 978-3-658-16243-6

[50] Schwering, M.: *Using Flink with MongoDB to enhance relevancy in personalization.* – URL https://2015.flink-forward.org/index.html%3Fp=400.html. – Zugriffsdatum: 2021-02-01

[51] Shellhammer, A. ; Neel, J.: *The Need for Mobile Speed.* – URL https://www.thinkwithgoogle.com/intl/de-de/marketing-strategien/apps-und-mobile/the-need-for-mobile-speed/. – Zugriffsdatum: 2021-03-16

[52] Statista: *Anzahl der Visits von amazon.de von März 2019 bis Oktober 2020.* – URL https://de.statista.com/statistik/daten/studie/995588/umfrage/anzahl-der-visits-pro-monat-von-amazonde/. – Zugriffsdatum: 2020-11-26

[53] Statista: *Prognose zur Anzahl der E-Commerce-Nutzer in Deutschland in den Jahren 2017 bis 2024.* – URL https://de.statista.com/prognosen/488012/prognose-der-e-commerce-nutzer-in-deutschland. – Zugriffsdatum: 2020-11-26

[54] Statista: *Umsatz durch E-Commerce (B2C) in Deutschland in den Jahren 1999 bis 2019.* – URL https://de.statista.com/statistik/daten/studie/3979/umfrage/e-commerce-umsatz-in-deutschland-seit-1999/. – Zugriffsdatum: 2020-11-26

[55] Streibich, K. ; Zeller, M.: Offene Plattformen als Erfolgsfaktoren für Künstliche Intelligenz. In: *Künstliche Intelligenz - Mit Algorithmen zum wirtschaftlichen Erfolg*. Springer Gabler, 2019. – ISBN 978-3-662-57567-3

[56] Stüber, E.: *Personalisierung im Internethandel - Die Akzeptanz von Kaufempfehlungen in der Bekleidungsbranche*. 2. Auflage. Springer Gabler, 2013. – ISBN 978-3-8349-4558-7

[57] von der Hude, M.: *Predictive Analytics und Data Mining - Eine Einführung mit R*. Springer Vieweg, 2020. – ISBN 978-3-658-30152-1

[58] Weber, F.: *Künstliche Intelligenz für Business Analytics - Algorithmen, Plattformen und Anwendungsszenarien*. Springer Vieweg, 2020. – ISBN 978-3-658-29772-5

[59] Westerkamp, C.: Datenschutz gemäß DSGVO im datengetriebenen Marketing – ein Überblick. In: *Data-driven Marketing - Insights aus Wissenschaft und Praxis*. Springer Gabler, 2020. – ISBN 978-3-658-29994-1

[60] Wirtz, B.: *Electronic Business*. 5. Auflage. Springer Gabler, 2016. – ISBN 978-3-658-10346-0

[61] Zaharia, S. ; Hackstetter, T.: Segmentierung von Onlinekäufern auf Basis ihrer Einkaufsmotive. In: *Dialogmarketing Perspektiven 2016/2017 - Tagungsband 11. wissenschaftlicher interdisziplinärer Kongress für Dialogmarketing*. Springer Gabler, 2017. – ISBN 978-3-658-16834-6

# Anhang

## A.1 Inhalt der CD-ROM

Dieser Arbeit liegt eine CD-ROM mit folgenden Dateien bei:

- **Bachelor_Thesis_Mareile_Beernink.pdf**: Diese Arbeit in PDF-Format.
- **Bachelor_Thesis_Mareile_Beernink.zip**: Der Quellcode des Prototypen.
- **README.md**: Informationen zum Prototypen.
- **Test_Log.xlsx**: Die Ausgabe der durchgeführten Testklasse.java

## A.2 Testfälle der Klasse Testklasse.java

# Testfälle

| Angebot /Empfehlung | Testfall_ID | Testbeschreibung | Erwartung einer Mitteilung | Ergebnis |
|---|---|---|---|---|
| **A1** | *TF_A1_TF01* | 1. Der Kunde (1112) hat den Newsletter bereits abonniert | NEIN | Erf. |
| | *TF_A1_TF02* | 1. Der Kunde (1111) hat den Newsletter noch nicht abonniert | JA | Erf. |
| | *TF_A1_TF03* | 1. Der Kunde (1111) hat innerhalb der letzten drei Minuten bereits ein Angebot (A1) erhalten<br>2. Der Kunde hat den Newsletter noch nicht abonniert | NEIN | Erf. |
| **A2** | *TF_A2_TF01* | 1. Der Bestandskunde (2044) ist unter 25 und schaut sich ein beliebiges Produkt an | NEIN | Erf. |
| | *TF_A2_TF02* | 1. Der Bestandskunde (2044) ist unter 25 und kauft ein beliebiges Produkt | NEIN | Erf. |
| | *TF_A2_TF03* | 1. Der Neukunde (2019) ist über 25 und merkt sich ein beliebiges Produkt | JA | Erf. |
| | *TF_A2_TF04* | 1. Der Neukunde (2100) ist unter 25 und merkt sich ein beliebiges Produkt | JA | Erf. |
| | *TF_A2_TF05* | 1. Der Bestandskunde (2011) ist über 25 und merkt sich ein Produkt | NEIN | Erf. |
| | *TF_A2_TF06* | 1. Der Bestandskunde (2044) ist unter 25 und merkt sich ein Produkt | JA | Erf. |
| | *TF_A2_TF07* | 1. Der Bestandskunde (2044) ist unter 25 und hat innerhalb der letzten drei Minuten bereits ein Angebot (A2) erhalten<br>2. Der Bestandskunde ist unter 25 und merkt sich ein Produkt | NEIN | Erf. |
| **A3** | *TF_A3_TF01* | 1. Der Kunde (3000) schaut sich vier mal das Produkt (31111) an | NEIN | Erf. |
| | *TF_A3_TF02* | 1. Der Kunde (3000) schaut sich vier mal das Produkt (31111) an<br>2. Der Kunde schaut sich ein mal das Produkt (11405) an | NEIN | Erf. |
| | *TF_A3_TF03* | 1. Der Kunde (3000) schaut sich vier mal das Produkt (31111) an<br>2. Der Kunde kauft das Produkt | NEIN | Erf. |
| | *TF_A3_TF04* | 1. Der Kunde (3000) schaut sich vier mal das Produkt (31111) an<br>2. Der Kunde schaut sich das Produkt erneut an | JA | Erf. |
| | *TF_A3_TF05* | 1. Der Kunde (3000) hat innerhalb der letzten drei Minuten bereits ein Angebot (A3) erhalten<br>2. Der Kunde schaut sich erneut fünf mal das Produkt (31111) an | NEIN | Erf. |
| **A4** | *TF_A4_TF01* | 1. Der Kunde (4006) betrachtet vier mal ein Standard-Produkt (11500)<br>2. Der Kunde merkt sich zwei mal ein Neu-Produkt (11411)<br>3. Der Kunde betrachtet vier mal ein Sale-Produkt (31200) | NEIN | Erf. |
| | *TF_A4_TF02* | 1. Der Kunde (4011) merkt sich vier mal ein Sale-Produkt (31200)<br>2. Der Kunde betrachtet zwei mal ein Sale-Produkt (21149)<br>3. Der Kunde betrachtet vier mal ein Neu-Produkt (11411) | JA | Erf. |
| | *TF_A4_TF03* | 1. Der Kunde (4011) hat innerhalb der letzten drei Minuten bereits ein Angebot (A4) erhalten<br>2. Der Kunde merkt sich vier mal ein Sale-Produkt (31200)<br>3. Der Kunde betrachtet zwei mal ein Sale-Produkt (21149)<br>4. Der Kunde betrachtet vier mal ein Neu-Produkt (11411) | NEIN | Erf. |
| | *TF_A4_TF04* | 1. Der Kunde (4027) betrachtet vier mal ein Neu-Produkt (11411)<br>2. Der Kunde merkt sich zwei mal ein Standard-Produkt (11500)<br>3. Der Kunde betrachtet vier mal ein Neu-Produkt (31119) | JA | Erf. |
| **A5** | *TF_A5_TF01* | 1. Das Produkt (11360) erhält eine Bestandsmenge von acht<br>2. Der Kunde (5004) merkt sich das Produkt (11360) | NEIN | Erf. |
| | *TF_A5_TF02* | 1. Das Produkt (11990) erhält eine Bestandsmenge von drei<br>2. Der Kunde (5004) betrachtet Produkt (11990) | JA | Erf. |
| | *TF_A5_TF03* | 1. Der Kunde (5004) hat innerhalb der letzten drei Minuten bereits eine Mitteilung (A5) zum Produkt (11990) erhalten<br>2. Der Kunde (5004) betrachtet Produkt (11990)<br>3. Das Produkt (11990) erhält eine Bestandsmenge von drei | NEIN | Erf. |
| | *TF_A5_TF04* | 1. Der Kunde (5004) merkt sich das Produkt (11360)<br>2. Das Produkt (11360) erhält eine Bestandsmenge von eins | JA | Erf. |
| **E1** | *TF_E1_TF01* | 1. Die Kundin (1906) betrachtet vier mal das Produkt (31199) der Kategorie "Bekleidung weiblich"<br>2. Die Kundin merkt sich zwei mal das Produkt (11588) der Kategorie "Schuhe weiblich" | JA | Erf. |
| | *TF_E1_TF02* | 1. Der Kunde (1925) betrachtet vier mal das Produkt (11470) der Kategorie "Schuhe männlich"<br>2. Der Kunde merkt sich zwei mal das Produkt (21634) der Kategorie "Accessoire männlich" | JA | Erf. |

| | | | | |
|---|---|---|---|---|
| | *TF_E1_TF03* | 1. Der Kunde (1941) merkt sich zwei mal das Produkt (11470) der Kategorie "Schuhe männlich"<br>2. Der Kunde betrachtet zwei mal das Produkt (21634) der Kategorie "Accessoire männlich"<br>3. Der Kunde betrachtet zwei mal das Produkt (31117) der Kategorie "Bekleidung männlich" | NEIN | Erf. |

| | | | | |
|---|---|---|---|---|
| **E2** | *TF_E2_TF01* | Der Kunde (2912) ist unter 30<br>1. Der Kunde merkt sich zwei mal das Produkt (11552) mit einem Wert von über 50 €<br>2. Der Kunde betrachtet sechs mal das Produkt (11670) mit einem Wert unter 50 € | JA | Erf. |
| | *TF_E2_TF02* | Der Kunde (2934) ist über 30<br>1. Der Kunde merkt sich zwei mal das Produkt (11552) mit einem Wert von über 50 €<br>2. Der Kunde betrachtet sechs mal das Produkt (11670) mit einem Wert unter 50 € | NEIN | Erf. |
| | *TF_E2_TF03* | Die Kundin (2926) ist unter 30<br>1. Die Kundin betrachtet sechs mal das Produkt (11431) mit einem Wert von über 50 €<br>2. Die Kundin betrachtet vier mal das Produkt (11677) mit einem Wert unter 50 € | NEIN | Erf. |

| | | | | |
|---|---|---|---|---|
| **E3** | *TF_E3_TF01* | 1. Die Kundin (3915) betrachtet sieben mal das Produkt (11431) mit einem Wert von über 50 €<br>2. Die Kundin betrachtet drei mal das Produkt (11677) mit einem Wert unter 50 € | JA | Erf. |
| | *TF_E3_TF02* | 1. Der Kunde (3920) betrachtet fünf mal das Produkt (11670) mit einem Wert unter 50 €<br>2. Der Kunde betrachtet fünf mal das Produkt (11552) mit einem Wert von über 50 € | NEIN | Erf. |

**** Die Logs der Tests befinden sich auf der beigefügten CD-ROM ****

## 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:

**Personalisierung im E-Commerce unter Verwendung von Realtime Streaming Analytics**

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 |
|---|---|---|