Ein Bild, das Text, Schrift, Logo, Grafiken enthält.

KI-generierte Inhalte können fehlerhaft sein.

Bachelorarbeit

Internationale Hochschule Fernstudium

Studiengang: B. Sc. Wirtschaftsinformatik

Die Vorteile der High-Code Entwicklung im Vergleich zur Low-Code Entwicklung

Evsin Rahmiev Matrikelnummer: 32105477

Akademiestr. 6

68159 Mannheim

Betreuungsperson: Prof. Dr. Thorsten Fröhlich 02.05.2025

Abstract

Im Bereich der Softwareentwicklung bietet sich neben der Möglichkeit, eine Software konventionell über High-Code Entwicklung mittlerweile auch die Option, über Low-Code zu entwickeln. Beide Ansätze haben unterschiedliche Vor- und Nachteile, je nachdem, was für Ziele hinter der Entwicklung stehen. Die vorliegende Arbeit hat die Aufgabe, zu beantworten, welche Vorteile die High-Code Entwicklung im Vergleich zu Low-Code anbietet. Zum Zweck der Arbeit und zur Beantwortung der Forschungsfrage wurde ein Case Study durchgeführt, bei der eine einfache Anwendung zum Deutschlernen einerseits über die etablierten Webentwicklungstechnologien HTML, CSS und JavaScript, also High-Code Werkzeuge, und andererseits über die Low-Code Entwicklungsanwendung *Zoho Creator* implementiert wurde. Es wurde untersucht, welche der beiden Entwicklungsansätze besser geeignet ist, um die konzipierte Lernanwendung mit ihren Funktionalitäten wie z. B. Verlinkung zwischen den Seiten der Anwendung, Manipulation der UI-Elemente, die grafische und funktionelle Anpassung von Formularen umzusetzen. Die Ergebnisse zeigen, dass sich die geplanten Anwendungskomponenten über High-Code Entwicklung näher an die Anforderungen implementieren lassen, weil diese Standardwebentwicklung mehr Anpassungsmöglichkeiten bietet. Über Low-Code hat man eher den Vorteil, dass man mit weniger Programmierkenntnissen eine Anwendung bauen kann, die dafür aber technisch weniger vielfältig ist.

Inhaltsverzeichnis

AbstractI

I. AbkürzungsverzeichnisII

II. AbbildungsverzeichnisIII

III. TabellenverzeichnisIV

1. Einleitung1

2. Theoretische Fundierung1

3. Methodik/Forschungsdesign2

4. Forschungsergebnisse3

4.1 4

5. Interpretation12

6. Fazit15

IV. Literaturverzeichnis16

I. Abkürzungsverzeichnis

1. H.C. – High-Code1

2. L.C. – Low-Code3

3. L.C. / N. C. – Low-Code / No-Code3

4. lDE – Integrated Development Environment2

5. HTML – Hypertext Markup Language1

6. CSS – Cascading Style Sheets1

7. DOM – Document Object Model1

8. OOP – Object Oriented Programming1

II. Abbildungsverzeichnis

Abb 1: Formel zur Berechnung des VI2

III. Tabellenverzeichnis

Tab: Projekt- und Releaseplan mit Iterationsraster, Phasen und Meilensteinen für Vorabnahmen, Releases und Abnahmen13

1. Einleitung

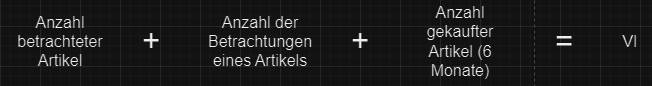
2. Gewährleistung von Preisnachlässen zur Absatzsteigerung

Damit das Nutzerverhalten der Kunden effektiv analysiert werden kann, müssen unterschiedliche Aspekte des Kundenverhaltens im Online-Shop per Clickstream betrachtet werden wie, u. a., die Anzahl der betrachteten Artikel, die Anzahl der Betrachtungen des gleichen Artikels und wie viele Artikel innerhalb der letzten 6 Monate gekauft wurden (Sakalauskas & Kricsciuniene, 2024, S. 2). Wenn diese effektiv berücksichtigt worden sind, können potenzielle Kunden identifiziert werden. Nach erfolgreicher Erhebung und Auswertung der Daten über das Kundenverhalten können die daraus gewonnenen Erkenntnisse folgendermaßen angewendet werden, um dann im Anschluss den Absatz zu steigern: individualisierte bzw. personalisierte Angebote in Form von Preisnachlässen können gewährleistet werden und zwar i. H. v. 30%. Die Erteilung solcher Preisnachlässe soll nur bei den Kunden stattfinden, die sich als wahrscheinlich zu kaufen oder als wiederholte Käufer (Kunden, die mindestens zwei Käufe beim gleichen Unternehmen tätigen) nachgewiesen haben (Sakalauskas & Kricsciuniene, 2024, S. 5). Es ist wichtig, diese Kunden zu identifizieren, damit die Werbungskosten und Kosten für Erstellung personalisierter Angebote in einer sinnvollen Höhe gehalten werden können (Sakalauskas & Kricsciuniene, 2024, S. 5).

3. Erweiterung der IT-Infrastruktur für die Analyse von Kundennutzungsdaten

Damit die bisher aufgezählten Messgrößen für die Verbindlichkeit und Ernsthaftigkeit des Kundenverhaltens erhoben werden können, muss die der Online-Shop ausgebaut werden. Sein Ausbau wird dafür sorgen, dass aus der Aktivität der Kunden abgeleitet werden kann, wie wahrscheinlich sie sind, Produkte zu kaufen, und somit eine Art Kundenzentriertheit („Customer Centricity“) erreicht wird (Hassler, 2021, S. 30). Damit dies festgestellt werden kann, wird der Wert der Messgröße Verbindlichkeitsindex (VI) ermittelt, welche die folgenden Kriterien einschließt: Anzahl der Artikel, die betrachtet wurden, Anzahl der Betrachtungen eines Artikels und Anzahl der Artikel, die innerhalb der letzten 6 Monate über den Shop gekauft wurden (Sakalauskas & Kricsciuniene, 2024, S. 5). Damit ein Besucher des Shops als wahrscheinlicher Käufer eingestuft wird, muss jedes Kriterium einen Wert von mindestens 1 aufweisen, also ein betrachteter Artikel, einmal den Artikel betrachtet haben und ein gekaufter Artikel in den letzten 6 Monaten:

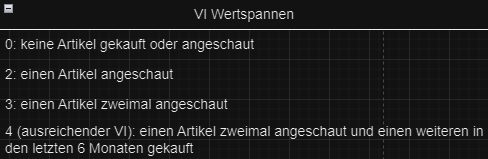
Abb 1: Formel zur Berechnung des VI



Quelle: Eigene Darstellung auf der Basis von Sakalauskas & Kricsciuniene, 2024, S. 5.

Alle drei Größen des VI ergeben jeweils einen Wert von 1 und die Anzahl der Betrachtungen eines Artikels verhalten sich proportional zu der ersten Betrachtung eines Artikels, da die Betrachtung eines Artikels automatisch dazu führt, dass man sich einen Artikel mindestens einmal angeschaut und somit einen VI von 2 hat. Die Werte aller Größen werden miteinander summiert und da für den VI kein maximaler Wert vordefiniert ist, kann er einen beliebig hohen Wert erreichen. Um allerdings als ernsthafter potenzieller Käufer zu gelten, muss ein Besucher des Shops einen VI Wert von mindestens 4 erreichen, wofür es mehrere unterschiedliche Kombinationen zum Erreichen dieses Wertes gibt: ein Besucher kann entweder sich ein Artikel insg. 3 Mal vor dem Kauf angeschaut haben, zwei Artikel angeschaut haben, sich ein Artikel zwei Mal angeschaut und in den vergangenen 6 Monaten einen weiteren Artikel gekauft haben, einen Artikel einmal angeschaut und in den letzten 6 Monaten 2 weitere Artikel gekauft oder sich in den letzten 6 Monaten einfach 4 Artikel gekauft haben, ohne in der aktuellen Session sich einen Artikel angeschaut zu haben.

Abb 2: Beispielhafte VI Wertspannen



Quelle: Eigene Darstellung

Die Ermittlung des VI wird durch die angesagte Erweiterung der Shop-Software realisiert. Dabei werden im Quellcode die entsprechenden Funktionen, sogenannte Event-Listener, eingesetzt, die bestimmte Ereignisse seitens des Nutzers wahrnehmen und entsprechend darauf reagieren (Stefanov & Sharam, 2021, S. 251). In diesem Fall werden Event-Listener verwendet, die das Klicken auf Artikellinks, die Anzahl der Betrachtungen, also wie viele Male auf dem Artikellink geklickt wurde, und Anzahl der Käufe in der Bestellhistorie erfassen können.

4. Geeignete Werkzeuge

Im vorherigen Kapitel wurden die Event-Listener als Mittel bekanntgegeben, welches zur Wahrnehmung des Nutzerverhaltens im Online-Shop verwendet wird. Diese Funktionen wurden ausgewählt, da der Online-Shop als Webanwendung mittels der Programmiersprache JavaScript umgesetzt wurde und die Event-Listener JavaScript-Funktionen sind, die auch erlauben, mit Elementen des DOM (Document Object Model) zu arbeiten (Stefanov & Sharam, 2013, S. 227). Das DOM stellt jede Webanwendung oder genauer gesagt, HTML-Anwendung, als ein Baum, der aus sogenannten „Nodes“ besteht, die die Äste des Baumes darstellen und somit dem Nutzer mittels Funktionen und Eigenschaften ermöglichen, auf Elemente aus dem DOM (Äste vom Baum) zuzugreifen, diese zu bearbeiten, entfernen oder neue hinzuzufügen (Stefanov & Sharam, 2013, S. 227).

Abb 3: HTML-Grundgerüst



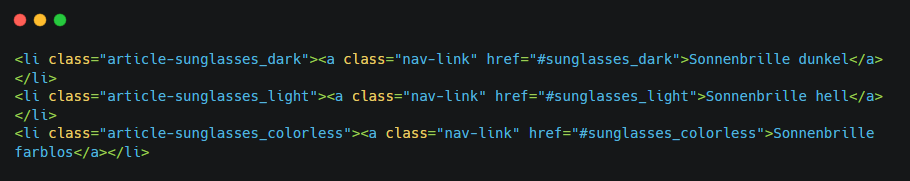
Quelle: Eigene Darstellung auf der Basis von Robbins, 2018, S. 56.

Nach dieser Erläuterung stellen die Bereiche des Onlineshops, welche das Nutzerverhalten verfolgen werden, Elemente bzw. Nodes des DOM, die grundsätzlich im HTML-Grundgerüst als Elemente angelegt und mithilfe der entsprechenden Event-Listener in der Lage sein werden, das Nutzerverhalten überhaupt wahrzunehmen.

4.1 Funktionen zur Analyse des Kundenverhaltens

Die Artikel, die sich ein Kunde als Besucher des Online-Shops anschauen oder auch kaufen kann, werden in der Codebasis als auswählbare Links innerhalb einer geordneten Liste angelegt:

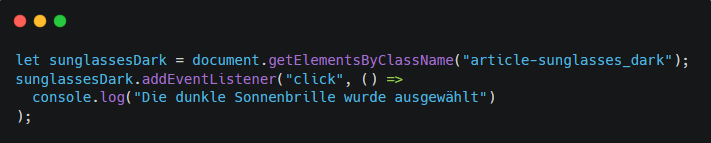
Abb 4: Sonnenbrillen-Artikel als auswählbare Links im HTML-Code



Quelle: Eigene Darstellung auf der Basis von Stefanov & Sharam, 2013, S. 227.

In diesem Falle verfügen alle Sonnenbrillen-Artikel über einen eigenen Bezeichner bzw. „class=“-Attribut, das mehreren Artikeln einen einzigartigen Bezeichner verleiht (Stefanov & Sharam, 2013, S. 227). Diese Bezeichner dienen dazu, dass sie in der Funktionslogik des Shops von dem jeweiligen Event-Listener, der für jede Sonnenbrille zuständig ist, erkannt werden. Die dunkle Sonnenbrille z. B. hat eine Event-Listener Funktion, die folgendermaßen aufgebaut ist:

Abb 5: Click-Funktion zum Artikel „Sonnenbrille dunkel“

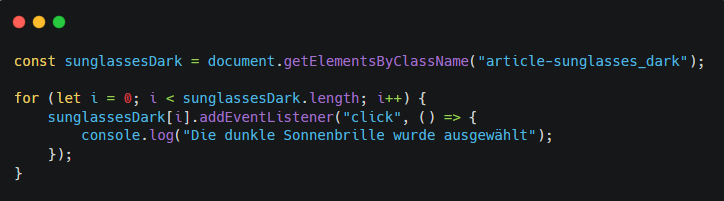


Quelle: Eigene Darstellung auf der Basis von Stefanov & Sharam, 2013, S. 252.

Die dunkle Sonnenbrille wird als Variable angelegt, die die Funktion „getElementByClassName()“ aufruft, um die den Artikel basierend auf seinen Bezeichner wiederum aufrufen zu können (Stefanov & Sharam, 2013, S. 237). Die Variable ruft im nächsten Schritt die „addEventListener()“-Funktion auf, die als Parameter einerseits das genaue Nutzerereignis bzw. Event und andererseits die Funktion übergeben bekommt, die sie beim Anklicken des Artikels ausführen soll (Stefanov & Sharam, 2013, S. 216). In diesem Falle soll die Funktion einfach einen Text in die Entwicklerkonsole ausgeben, der darauf hinweist, welcher Artikel ausgewählt wurde.

Wie oft ein Artikel von einem Kunden betrachtet wurde, lässt sich durch eine leichte Erweiterung des Event-Listeners, der auf das Anklicken eines Artikel-Links achtet, umsetzen:

Abb 6: Zählen der Artikelaufrufe

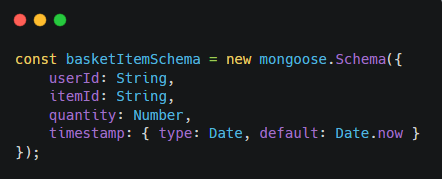


Quelle: Eigene Darstellung auf der Basis von Stefanov & Sharam, 2013, S. 252.

Die hier vorgenommene Änderung zum initialen Event-Listener führt die Funktion innerhalb einer for-Schleife aus, eine Kontrollstruktur, die uns erlaubt, in jeweils jedem Schleifen-Durchlauf einen Vorgang auszuführen (Stefanov & Sharam, 2013, S. 217). In diesem Fall stellt jedes Anklicken eines Artikellinks einen Durchlauf bzw. eine Iteration dar und soll dazu führen, dass die Anzahl der Klicks auf dem Artikel-Link iteriert, also erhöht werden, und bei jedem Klick der Hinweistext in die Entwickler-Konsole ausgegeben wird.

Damit das Verfolgen der Anzahl der Artikel, die innerhalb der letzten 6 Monate gekauft wurden, möglich ist, müssen bei jedem Kauf die zugehörigen Daten in der Datenbank, die an dem Shop gebunden ist, gespeichert werden. In dem Fall der gekauften Artikel wird dies, in Anlehnung an Copeland (2013, S. 19) durch ein polymorhpisches Schema verwirklicht, die den Nutzerbezeichner („userId“), Artikelbezeichner („itemId“), Menge („quantity“) und Zeitstempel („timestamp“) einschließt:

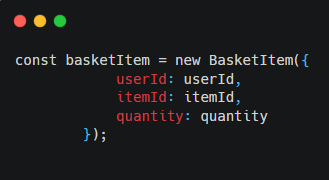
Abb 7: Polymorphisches Datenschema eines Artikelkaufs



Quelle: Eigene Darstellung auf der Basis von Copeland, 2013, S. 19.

Das polymorphische Datenschema ermöglicht, dass serverseitig alle Datensätze in dem gleichen Datentyp gespeichert werden, unabhängig davon, welche oder wie viele Daten in einem Datensatz vorhanden sind (Copeland, 2013, S. 19). Im Gegensatz zu dem NoSQL- bzw. nicht relationalem Datenbanksystem MongoDB (Meier, 2018, S. 9) wäre bei der Nutzung einer relationalen Datenbank das Anlegen mehrerer unterschiedlicher Datenschemas notwendig, um das Speichern von jedem Typ Datensatz abhängig davon, genau welche Daten in dem Datensatz vorhanden sind, zu ermöglichen (Copeland, 2013, S. 19). Darauf basierend wird bei jedem Kauf eines Kunden ein neuer Datensatz in der Datenbank angelegt:

Abb 8: Anlegen eines Artikeldatensatzes in der Datenbank



Quelle: Eigene Darstellung auf der Basis von Copeland, 2013, S. 19.

Die Überprüfung der Käufe der letzten 6 Monate erfordert eine Variable, in der die Zahl 6 von dem Wert des aktuellen Monats subtrahiert wird. Daraufhin wird per asynchrone Funktion (Stefanov & Sharam, 2013, S. 260) herausgefunden, welche Käufe bzw. BasketItems zu einem Benutzer für den Zeitraum der letzten 6 Monate gefunden werden können und das Ergebnis wird entsprechend zurückgegeben:

Abb 9: Prüfen der Käufe aus den letzten 6 Monaten

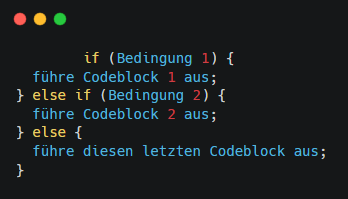


Quelle: Eigene Darstellung in Anlehnung an Stefanov & Sharam, 2013, S. 260.

4.2 Ermittlung des VI

Eine if-Verzweigung wird dafür sorgen, dass aus dem Nutzerverhalten abgeleitet wird, was für ein VI jeder Nutzer innerhalb einer Session aufweisen kann. Eine solche Verzweigung prüft, ob eine Bedingung, die in den Klammern nach dem Schlüsselwort „if“ erfüllt ist und falls ja, führt den ersten Codeblock aus – als Codeblock wird der Code bezeichnet, der sich innerhalb eines eckigen Klammerpaares ( {Codeblock} ) befindet (Flanagan, 2020, S. 102). Falls die Bedingung nicht erfüllt ist, kann die Verzweigung prüfen, ob die „else if“-Bedingung erfüllt ist und falls ja, führt den Codeblock dazu aus. Falls keine dieser Bedingungen erfüllt ist, wird der Codeblock in nach der „else“-Anweisung ausgeführt:

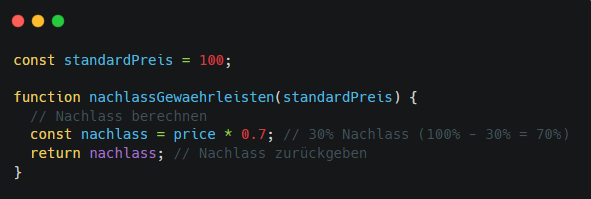
Abb 10: If-Verzweigung



Quelle: Eigene Darstellung in Anlehnung an Flanagan, 2020, S 102.

Eine if-Verzweigung kann aus beliebig vielen Bedingungen bestehen. Die if-Verzweigung für den Anwenderfall des Online-Shops wird neben der Prüfung des Kundeverhaltens auch dafür sorgen, dass die Kunden, die einen VI im Wert von 4 erreicht haben, einen pauschalen Preisnachlass i. H. v. 30 % vom Gesamtwert des Warenkorbs innerhalb der aktuellen Session bekommen. Für die Implementierung wird eine Variable deklariert und initialisiert, die die Höhe des Standardpreises festlegt und eine Funktion, die den Nachlass berechnet und ihn zurückgibt:

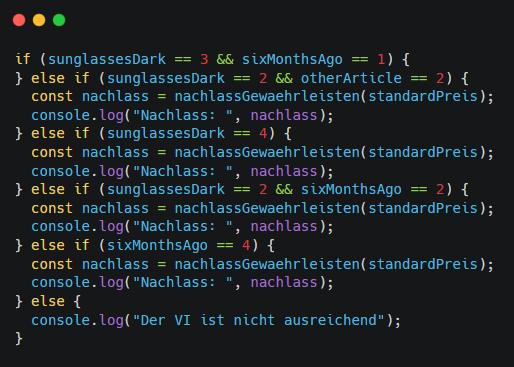
Abb 11: Standardpreisvariable und Nachlassberechnungsfunktion



Quelle: Eigene Darstellung.

Die Berechnungsfunktion wird im nächsten Schritt in der if-Verzweigung in jedem Codeblock eingesetzt, der prüfen soll, welche Konstellation vorhanden ist, die einem VI von 4 entspricht:

Abb 12: Gewährleistung vom Preisnachlass bei ausreichendem VI



Quelle: Eigene Darstellung auf der Basis von Flanagan, 2020, S. 101.

Solange eine Konstellation erfüllt und somit VI = 4 erreicht wurde, wird die Funktion ausgeführt und der Kunde bekommt einen Nachlass. Falls keine der Konstellationen erfüllt ist, wird die Meldung „Der VI ist nicht ausreichend“ in die Konsole ausgegeben.

4.3 Erwartete Ergebnisse

Bereits durchgeführte Forschungen weisen auf die Effektivität von der Clickstream Analyse in Kombination mit personalisierten Angeboten und ihren Beitrag zur Absatzsteigerung. Eine von der NAI durchgeführte Studie wird hier kurz in Form von Kuchendiagrammen erläutert und später bei der Projektvorstellung genauso vorgestellt, weil auf einem Kuchendiagram Zahlen einfacher vom Leser zu erfassen und zu lesen sind (Beck, 2014, S. 135). NAI (2010, S. 2) hat festgestellt, dass bis zu 18% des Werbungsumsatzes von personalisierten Angeboten stammt:

Abb 13: Umsatz durch Werbung

Quelle: Eigene Darstellung in Anlehnung an NAI, 2010, S. 2.

Ferner hat die NAI herausgefunden, dass bis zu 91% der Kunden sich für einen Anbieter entscheiden würden, der ihnen personalisierte Angebote unterbreitet:

Abb 14: Einkaufsbereitschaft bei personalisierten Angeboten

Quelle: Eigene Darstellung in Anlehnung an NAI, 2010, S. 2.

Von diesen 91% der Einkäufer würden 49% mehrfach bei dem gleichen Anbieter einkaufen:

Abb 15: Mehrfache Kaufbereitschaft bei personalisierten Angeboten

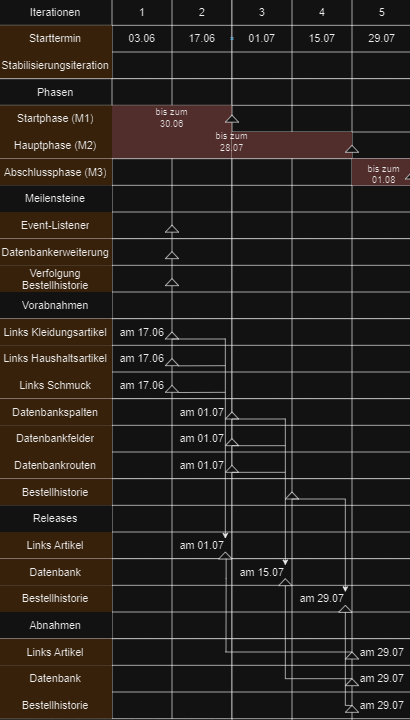
Quelle: Eigene Darstellung in Anlehnung an NAI, 2010, S. 2.

Die Ergebnisse der NAI Studie deuten darauf hin, dass verhaltensbasierte personalisierte Angebote auch bis zu ein Fünftel des Umsatzes der Global Retail AG ausmachen können, der durch Werbung erzielt wird, und einen Großteil der Online-Einkäufern sichern und viele dieser Kunden dazu bringen kann, Stammkunden der Global Retail AG zu werden.

5. Projektplan

Für die Umsetzung der bisher erläuterten Maßnahmen zur Absatzsteigerung wird im ersten Schritt innerhalb eines Meetings die Implementierungsidee als Projektplan vorgestellt. Im Meeting wird die Anwesenheit der relevanten Stakeholder erforderlich. Dazu zählen die Geschäftsleitung der Global Retail AG, die Vertriebsleitung und das Entwicklungsteam. Die Notwendigkeit des Projekts wird anhand der NAI-Studie argumentiert und die Verfolgung des Kundenverhaltens mithilfe des Ausbaus der Codebasis des Online-Shops erklärt. Es wird ein Umsetzungszeitraum von zwei Monaten prognostiziert, der nach der Genehmigung der Geschäftsführung anfängt. Der vorgestellte Quellcode zur Umsetzung der Event-Listener Funktionen wird dem Leiter der Entwicklungsabteilung weitergegeben, der damit beauftragt wird, die Umsetzung der Funktionen im Code, der Datenbankerweiterungen und das Testen der neuen Funktionalitäten als Tickets anzulegen, die innerhalb des Entwicklungsteams je nach Erfahrung und Verfügbarkeit der Entwickler verteilt werden. Die Tickets werden über das Verwaltungstool Jira vom Softwarehersteller Atlassian verwaltet, damit „Informationen […] zwischen den Workflows sicher ausgetauscht werden können“ (Bayer, 2021, S. 1). Jedes Ticket bekommt eine Überschrift, Aufgabenbeschreibung, Priorisierung, Angabe der Ticketart und wird an die zuständigen Entwickler zugewiesen. Die Tickets werden auch je nachdem verteilt, welche Entwickler im Frontend- und welche im Backend-Bereich spezialisiert sind. Die Frontend-Entwickler werden für die Umsetzung der Event-Listener Funktionen und die Backend Entwickler für den Aufbau der Routen zur Datenbank zuständig sein. Jeder Entwickler wird eine eigene Branch des Quellcodes erstellen, die auf einer Kopie der auf einer Repository, in diesem Falls GitHub, gespeicherten Codebasis basieren wird (Tsitoara, 2024, S. 150). Innerhalb eines Daily Standups wird der Stand der Implementierung innerhalb der Entwicklungsabteilung besprochen, wobei der Abteilungsleiter als Moderator fungieren wird (Mauerhofer, 2024, S. 18). Es findet ein agiles Projektmanagement nach dem APM-Verfahren statt, bei welchem die Projektlaufzeit als iterative Vorgehen in eine Sequenz von Zeitfenstern eingeteilt wird, die man Iterationen nennt (Oestereich, Weiss, Lehmann & Vigenschow, 2014, S. 3). Es wird agil gearbeitet, weil die Prinzipien der agilen Softwareentwicklung sicherstellen, dass die Software inkrementell und in kurzen Iterationen erstellt wird, Fachexperten und Entwickler möglichst direkt und täglich zusammenarbeiten und das Entwicklungsteam in regelmäßigen Abständen darüber reflektiert, wie es die gemeinsame Arbeit verbessern kann (Oestereich, Weiss, Lehmann & Vigenschow, 2014, S. 18). Der Projektablauf wird in Phasen Meilensteinen für Vorabnahmen, Releases und Abnahmen eingeteilt und anhand einer Tabelle, die einer Gantt-Chart ähnelt, visualisiert (Oestereich, Weiss, Lehmann & Vigenschow, 2014, S. 8). Im Projektplan werden im Tabellenkopf die Termine für die Iterationen angegeben. Das Projekt besteht aus den Phasen Startphase, Hauptphase und Abschlussphase und auf dem Projektplan bzw. der Projekttabelle wird verdeutlicht, bis zu welchem Termin die jeweiligen Projektphasen abzuschließen sind:

Tab: Projekt- und Releaseplan mit Iterationsraster, Phasen und Meilensteinen für Vorabnahmen, Releases und Abnahmen.



Quelle: Eigene Darstellung (geändert) in Anlehnung an Oestereich, Weiss, Lehmann & Vigenschow, 2014, S. 7.

Die Meilensteine stellen die großen Arbeitspakete dar, die die inhaltlich zusammengehörenden Teilaufgaben einschließen. Innerhalb der Vorabnahmen werden die Implementierungen der Entwicklungs- und Projektleitung vorgestellt, um den Fortschritt zu präsentieren und die Ergebnisse mit den Erwartungen abzugleichen. Bei dem Release werden die Implementierungen in das Live-System übertragen und bei der Abnahme den anderen relevanten Stakeholdern, also der Geschäftsleitung und Vertriebsabteilung vorgestellt.

6. Schluss

IV. Literaturverzeichnis

Baumgarten, C., Rainer, E. & Stich, S. (2024): Professionelle Softwareentwicklung mit Low Code optimieren – eine Fallstudie. In: *HMD Praxis der Wirtschaftsinformatik*. Springer Verlag.

Brandt-Pook, H. & Kollmeier, R. (2020): *Softwareentwicklung kompakt und verständlich. Wie Softwaresysteme entstehen*. (3. Auflage). Springer Verlag.

Böhler, T. (2023): Software einfacher, flexibler und leichtfüßiger entwickeln. In: *Produktion*. (12):16.

Elshan, E. & Binzer, B. (2024): Mehr als ein Trend?: Wie Low-Code die digitale Transformation unterstützt. In: *HMD Praxis der Wirtschaftsinformatik*. 1070-1087.

Frank, B. (2023): *ERP:* The next big thing: Sind Lösungen auf Low-Code-Basis die Zukunft? In: *IT-Management*. 50-54.

Hensen, U. (2023): Maßgeschnittene Software mittels Low Code. In: *Factory Innovation*. 33-37.

Schreiner, K. (2024): Entwicklungszyklen halbieren mit Low Code. In: *Digital Engineering*. 44-45.

Siroker, D. & Koomen, P. (2013): *A/B Testing. The Most Powerful Way to Turn Clicks into Customers*. Wiley.

Spierling, D. (2023): Mehr IT-Nachhaltigkeit und Klimaschutz durch Low-Code Development. In: *Wirtschaftsinformatik & Management*. 107–113.

Eidesstattliche Erklärung

Ein Bild, das Text, Schrift, Logo, Grafiken enthält.

KI-generierte Inhalte können fehlerhaft sein.

Hiermit versichere ich an Eides statt, dass ich die Abschlussarbeit selbständig und ohne Inanspruchnahme fremder Hilfe angefertigt habe. Ich habe dabei nur die angegebenen Quellen und Hilfsmittel verwendet und die aus diesen wörtlich oder inhaltlich entnommenen Stellen als solche kenntlich gemacht. Die Arbeit hat in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegen. Ich erkläre mich damit einverstanden, dass die Arbeit mit Hilfe eines Plagiatserkennungsdienstes auf enthaltene Plagiate überprüft wird.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Ort, Datum Unterschrift