Probe-IPA 2023

Altersüberprüfung

Inhaltsverzeichnis

[Inhaltsverzeichnis 1](#_Toc129969334)

[Kapitel 1: Umfeld und Ablauf 2](#_Toc129969335)

[1 Aufgabenstellung 2](#_Toc129969336)

[2 Projektaufbauorganisation 3](#_Toc129969337)

[3 Vorkenntnisse 3](#_Toc129969338)

[4 Vorarbeiten 3](#_Toc129969339)

[5 Projektmanagementmethode 4](#_Toc129969340)

[6 Einrichtung des Arbeitsplatzes 4](#_Toc129969341)

[7 Versionierung und Sicherung 5](#_Toc129969342)

[8 Taskliste 6](#_Toc129969343)

[9 Zeitplan 7](#_Toc129969344)

[10 Arbeitsprotokoll 8](#_Toc129969345)

[Kapitel 2: Projekt 18](#_Toc129969346)

[1 Kurzfassung des IPA-Berichtes 18](#_Toc129969347)

[2 Informieren 19](#_Toc129969348)

[3 Planen 21](#_Toc129969349)

[4 Entscheiden 32](#_Toc129969350)

[5 Realisieren 33](#_Toc129969351)

[6 Kontrollieren 44](#_Toc129969352)

[7 Auswerten 45](#_Toc129969353)

[Inhaltsinformationen 46](#_Toc129969354)

[1 Abkürzungsverzeichnis 46](#_Toc129969355)

[2 Glossar 46](#_Toc129969356)

[3 Quellenverzeichnis 48](#_Toc129969357)

[4 Abbildungen 49](#_Toc129969358)

[5 Tabellen 50](#_Toc129969359)

[6 Erläuterung zum Anhang 51](#_Toc129969360)

Kapitel 1: Umfeld und Ablauf

# Aufgabenstellung

## Ausgangslage

Die Inteco EDV AG setzt für Getränkehändler Webshops mit Anbindung an das eigene ERP-System WEGAS um. Da viele dieser Kunden alkoholhaltige Getränke verkaufen, muss laut den neuen Onlinegesetzen der Schweiz eine Altersüberprüfungsmethode im Webshop implementiert werden.

Vom Kunden «Wieland AG» Wurde der Auftrag vermittelt, eine Altersüberprüfung in ihrem Webshop einzubauen, welche durch die Kennzahlen der Schweizer ID oder des Schweizer Passes erfolgen soll.

Die Webseite des Kunden ist ein «Standartshop», der Auftrag sollte aber auch für die «CustomView Shops» umgesetzt werden.

## Detaillierte Aufgabenstellung

Im Dialog der Altersüberprüfung sollte man die Kennzahlen in klar ersichtliche und entsprechende Felder einfügen können. Dabei kann man sich an dem von [galaxus.ch](https://www.galaxus.ch/) verwendeten Design stützen.

Eine Bestellung kann nur abgeschlossen werden, falls das Alter überprüft wurde. Es ist zu unterscheiden zwischen Ü16 und Ü18.

Falls der Benutzer eingeloggt ist, sollte die Überprüfung in einem neuen Datensatz auf dem Konto hinterlegt werden. Es dürfen jedoch keine Ausweisdaten gespeichert werden.

Für den Endbenutzer muss es die Option geben, diese Verifizierungsart abwählen und eine manuelle Überprüfung anzufordern. Falls die Überprüfung fehlschlägt oder der Kunde die Angaben nicht abfüllen will oder keine Schweizer ID/Pass hat soll eine Kontaktadresse des Shop Inhabers angezeigt werden, damit dieser die Überprüfung manuell tätigen kann.

Demnach braucht es im Magnolia Kunden-Menü auch die Möglichkeit die Ü16/18 Flags für Kundenkonten manuell zu hinterlegen.

## Mittel und Methoden

* HTML
* Freemarker
* SCSS/CSS
* JavaScript
* JQuery
* Java
* Magnolia
* Apache Tomcat 7
* Eclipse
* Visual Studio Code

# Projektaufbauorganisation

Tabelle Projektaufbauorganisation

|  |  |  |
| --- | --- | --- |
| Name, Vorname | Kontakt | Funktion |
| Chirokikh Shevoroshkin, Georgiy | [georgiy.shevoroshkin@inteco.ch](mailto:georgiy.shevoroshkin@inteco.ch)  077 215 64 36 | Kandidat |
| Hiestand, Betina | [betina.hiestand@inteco.ch](mailto:betina.hiestand@inteco.ch)  055 533 00 64 | Verantwortliche Fachkraft |
|  |  | Hauptexperte |
|  |  | Nebenexperte |

# Vorkenntnisse

Ich besuchte drei Jahre lang die Informatikmittelschule Hottingen und bin seit einem halben Jahr bei der Inteco EDV AG als Applikationsentwickler angestellt. Ich habe bereits Erfahrung in folgenden Bereichen gesammelt:

Tabelle Vorkenntnisse

|  |  |  |
| --- | --- | --- |
| HTML | Mehrere Projekte damit realisiert | Advanced |
| SCSS/CSS | Mehrere Projekte damit realisiert | Advanced |
| JavaScript | Mehrere Projekte damit realisiert | Intermediate |
| JQuery | Mehrere Projekte damit realisiert | Intermediate |
| CMS Magnolia | Tägliche Anwendung seit 6 Monaten | Beginner |
| Java | Tägliche Anwendung seit 6 Monaten, Schulkenntnisse | Intermediate |
| Freemarker | Tägliche Anwendung seit 6 Monaten | Beginner |
| Apache Tomcat 7 | Tägliche Anwendung seit 6 Monaten | Beginner |
| Windows 10 | Tägliche Benutzung | Intermediate |
| Eclipse | Tägliche Anwendung seit 6 Monaten, Schulkenntnisse | Intermediate |
| Visual Studio Code | Tägliche Benutzung | Advanced |
| Microsoft Word | Tägliche Benutzung | Advanced |
| Microsoft Excel | Tägliche Benutzung | Intermediate |
| Microsoft Visio | Grundkenntnisse aus der Schule | Beginner |

# Vorarbeiten

Vorlage des Berichtes.

Aufsetzen einer lokalen dmvini-Webapp.

* Kopieren einer Standartshop-Webapp
* Erstellen eines Shop-Branch (shop-ageCheck-gs2)
* Tomcat 7 Server erstellen und Webapp einbinden
* Initialen Installationsdurchgang lokal durchführen
* Lokaler Restore eines Backup des live Shops

Es wurde die Dmvini-Webapp gewählt, da es keine Wieland Testinstanz, jedoch eine Dmvini Testinstanz gibt, um später in der Kontroll-Phase die Umsetzung auf einer Online-Webapp zu testen. Da beide Webapps Standartshops sind, kann die Funktionalität im Shop-Modul entwickelt werden und für alle Standartshops verwendet werden.

# Projektmanagementmethode

Ich entschied mich für meine IPA die IPERKA Methode anzuwenden. Ich empfinde es für sinnvoll auf diese Art das Projekt durchzuführen, da es dadurch von Grund auf gut geplant und verstanden, somit auch effizienter realisiert werden kann. Meiner Meinung nach eignet sich das IPERKA Modell für die IPA vor allem aus dem Grund gut, dass die Planungsphase stark gewichtet wird. Ausserdem bin ich mit diesem Vorgehen schon seit Zeiten der Schule vertraut, was mir die Durchführung der Arbeit auf diese Weise erleichtert. Im untenstehenden Bild werden die einzelnen Schritte von IPERKA veranschaulicht. (siehe Abbildung 1)

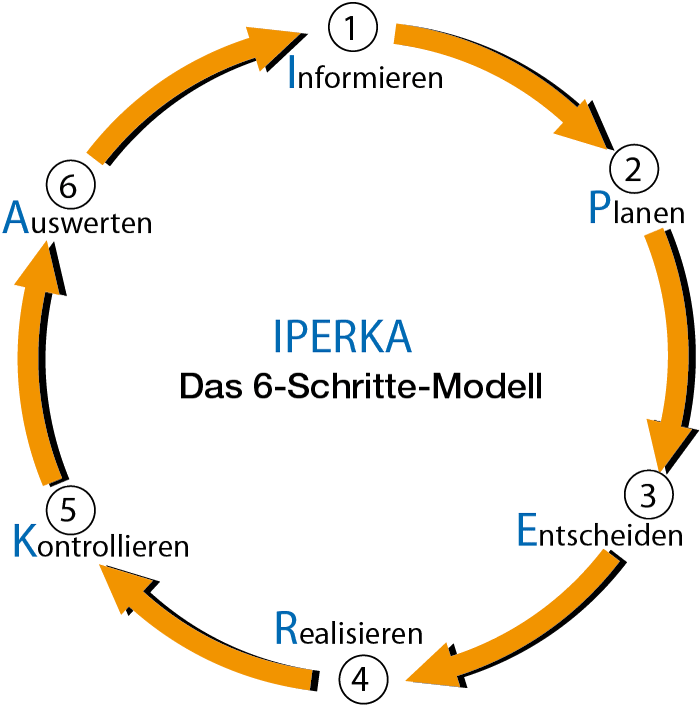


Abbildung IPERKA-Modell

# Einrichtung des Arbeitsplatzes

Während der IPA entschied ich mich aus mehreren Gründen, im Homeoffice zu arbeiten. Meine verantwortliche Fachkraft und die meisten Arbeitskollegen arbeiten auch von zuhause aus, also sehe ich keinen Vorteil im Büro zu sein in Bezug auf die Kommunikation, denn diese erfolgt sowieso per Teams. Da ich ADHS habe, brauche ich mehr Pausen zwischendurch, um effizienter arbeiten zu können. Ich kann die Zeit, die der Arbeitsweg einnimmt, genau dafür nutzen. Ausserdem ist meine Konzentration höher, je weniger Störende Einflüsse von aussen kommen. Deswegen gelingt es mir besser, allein in meinem Büro zu arbeiten. Dieses ist mit Tastatur, Maus und einem weiteren Bildschirm ausgerichtet, um mir den Workflow zu vereinfachen.

# Versionierung und Sicherung

## Versionierung

Zur Versionierung verwenden wir in der Inteco SVN, was ich auch während dieser IPA brauchen werde. Ich werde in meinem separaten Branch (shop-ageCheck-gs2) arbeiten und sobald ein Task erledigt ist, einchecken. Somit habe ich die Möglichkeit, auf ältere Versionen zugreifen zu können.

Die IPA wird täglich auf ein NAS in Pfäffikon Zürich über das Backup-Programm Veeam gesichert.

## Dokumentablage

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Abbildung Ordnerstruktur

# 

# Taskliste

Ein Bild, das Tisch enthält.

Automatisch generierte Beschreibung

Abbildung Taskliste

# Zeitplan

Ein Bild, das Text, Shōji enthält.

Automatisch generierte Beschreibung

Abbildung Zeitplan

# Arbeitsprotokoll

## Mittwoch, 01.03.2023

Tabelle Arbeitsprotokoll - Tag 1

|  |  |
| --- | --- |
| Ziele | Informieren   1. Berechnung 2. Galaxus 3. IPA Info 4. Dokumentation |
| Erreichte Ziele | Informieren   1. Berechnung 2. Galaxus 3. IPA Info 4. Dokumentation   Planen   1. Taskliste 2. Zeitplan |
| Aufgetretene Probleme | Keine |
| Beanspruchte Hilfe | Keine |
| Zeitplan | Im Zeitplan, mit 2h Überzeit |

### Workflow

Begonnen habe ich damit, mir einen Groben Überblick über den Auftrag zu beschaffen und Informationen zu sammeln. Gleichzeitig notierte ich mir die Aufgaben, in welche ich den Auftrag unterteilen würde. Ich fand einen weiterführenden [Webseiteneintrag](https://www.kleemans.ch/identit%C3%A4tskarte-berechnung-pr%C3%BCfziffer) zum Thema Alterskalkulation durch gegebene Prüfziffern einer Schweizer ID oder eines Schweizer Passes. Auch über die Umsetzung der Altersüberprüfung von [Galaxus](https://www.galaxus.ch/) wurde ich fündig und speicherte eine Bildschirmaufnahme davon.

Dann fing ich an, den ersten Teil des IPA-Berichtes zu schreiben. Als ich ein Grobkonzept davon hatte, vervollständigte ich die Taskliste und kreierte daraus einen Zeitplan.

### Reflexion

Heute war ich sehr produktiv und kam gut voran. Von der letzten probe-IPA nahm ich mit, dass ich mehr Zeit für die Dokumentation einplanen sollte und die Informations- und Planungsphase auf keinen Fall vernachlässigen sollte. Dies setzte ich heute auch um und da ich mir keinen grossen Zeitdruck mache und die Bedeutsamkeit der ersten Phasen der IPERKA Methode mir verinnerlichte, war ich aufnahmefähiger und konzentrierter. Ich habe dadurch schon jetzt eine ziemlich klare Vorstellung der Umsetzung der Arbeit. Ein weiterer Grund dafür kann aber auch sein, dass dieser Auftrag mir nicht so komplex erscheint.

## Donnerstag, 02.03.2023

Tabelle Arbeitsprotokoll - Tag 2

|  |  |
| --- | --- |
| Ziele | Planen   1. UseCase 2. Weitere Diagramme 3. Programmierplan 4. Testfälle |
| Erreichte Ziele | Planen   1. UseCase 2. Weitere Diagramme 3. Programmierplan 4. Testfälle 5. Beginn Dokumentation |
| Aufgetretene Probleme | Keine |
| Beanspruchte Hilfe | Betina Hiestand: Feedback Testkonzept und UseCase |
| Zeitplan | Im Zeitplan, mit 2h Überstunden |

### Workflow

Heute habe ich mit dem UseCase Diagramm begonnen, welches mir ein bisschen Schwierigkeiten bereitete, da ich nicht genau wusste, welche Prozesse dort reingehören und welche nicht. Deswegen bat ich Betina Hiestand drum, mir ein Feedback zum erstellten UseCase Diagramm zu geben.

Danach entwarf ich ein Flussdiagramm und die Testszenarien. Gleichzeitig schrieb ich die Fortschritte in den IPA-Bericht.

Ich merkte, dass in der Aufgabenstellung nicht definiert war, welche Produkte auf welches Alter überprüft werden sollten und woher diese Information geholt werden sollte. Deswegen plane ich, im JCR im Magnolia zu den vorhandenen Kategorien eine Mindestalter-Property einzufügen (zB Spirituosen = 18, Bier= 16).

### Reflexion

Wie schon erwähnt, war ich mich beim UseCase Diagramm unsicher. Daraus schliesse ich, dass ich mich mehr damit befassen soll, bevor ich die offizielle IPA starten werde.

Heute merkte ich, dass meine Konzentration nicht gleich vorhanden war wie gestern, was dazu geführt hat, dass ich meinem Dopaminsuchenden Gehirn nachgegeben hatte und die Aufgaben zuerst erledigte, die mehr Spass machten. In diesem Fall war das nicht sehr behindernd in Aussicht auf das Planen und Abschliessen des Gesamtprojekts, doch ich muss aufpassen, dass es nicht dazu kommt, dass ich unorganisiert irgendwelche Tasks erledige und daraus ein Chaos entsteht. Zum Glück kann ich das durch den Programmierplan hindern.

## Freitag, 03.03.2023

Tabelle Arbeitsprotokoll - Tag 3

|  |  |
| --- | --- |
| Ziele | Planen   1. Dokumentation |
| Erreichte Ziele | Planen   1. Dokumentation   Entscheiden   1. Entsheiden |
| Aufgetretene Probleme | Keine |
| Beanspruchte Hilfe | Keine |
| Zeitplan | Im Zeitplan, mit 4h Überstunden |

### Workflow

Ich habe damit begonnen, das UseCase und die Testszenarien zu dokumentieren. Zwischendurch beschaffte ich mir einen Überblick von den verschiedenen Dateien, die ich im Schritt Realisieren implementiert werden müssen. Es fielen mir während des Prozesses zwei Entscheidungen auf, die ich treffen musste. Da ich frühzeitig mit der Dokumentation fertig war, machte ich mir schon Gedanken um die nächste Phase.

### Reflexion

Da die Dokumentation oftmals Fleissarbeit ist, hatte ich manchmal Mühe, mich über längere Zeit damit zu befassen. Ich konnte jedoch meine Konzentration über den ganzen Tag beibehalten, indem ich die Aufgaben, an denen ich arbeitete, abwechselte. Als Beispiel schrieb ich eine Stunde die Dokumentation, befasste mich dann eine halbe Stunde damit, den Plan für die Realisationsphase zu erstellen, und begab mich danach wieder zum vorherigen Task.

## Montag, 06.03.2023

Tabelle Arbeitsprotokoll - Tag 4

|  |  |
| --- | --- |
| Ziele | Entscheiden   1. Dokumentation   Realisieren   1. ftl 2. java |
| Erreichte Ziele | Entscheiden   1. Dokumentation   Realisieren   1. ftl 2. java |
| Aufgetretene Probleme | Keine |
| Beanspruchte Hilfe | Keine |
| Zeitplan | Im Zeitplan, mit 4h Überstunden |

### Workflow

Ich habe als erstes die getroffenen Entscheidungen dokumentiert und ging dann weiter zum Realisieren des Auftrags. Diese Phase des Projekts begann ich damit, ein HTML Grundgerüst im FTL zu erstellen. Danach erstellte ich die Java-Klasse und schrieb die benötigten Methoden (ohne Logik) rein. Dazu erstellte ich je eine JS und CSS Datei. All das habe ich schlussendlich ins SVN committed.

Nach dem initialen commit habe ich im JS File eine JQuery Methode erstellt die nur die Ausgewählte Option zu Altersverifikation einblendet. Schlussendlich erstellte ich im Magnolia im JCR die benötigten Konfigurationen und Verknüpfungen, um die Komponente auf der Bestellübersichtsseite einzubinden.

### Reflexion

Ich kam heute gut voran und konnte dank dem erstellten Plan geordnet vorgehen. Das ermöglichte mir eine ruhigere und einfachere Ausführung der Aufgaben.

## Mittwoch, 08.03.2023

Tabelle Arbeitsprotokoll - Tag 5

|  |  |
| --- | --- |
| Ziele | Realisieren   1. Basketrule 2. Magnolia 3. scss |
| Erreichte Ziele | Realisieren   1. Java fertigstellen 2. Basketrule 3. Magnolia 4. Beginn Dokumentation |
| Aufgetretene Probleme | JavaScript/Java |
| Beanspruchte Hilfe | Loris Sigrist |
| Zeitplan | Im Zeitplan mit 4h Überstunden |

### Workflow

Begonnen habe ich damit, die ConfigPageIntern und shopUserPage umzusetzen, was mir gut gelungen ist. Anschliessend erstellte ich einen neuen Node in der Produktimport Konfiguration, um die Überkategorien, bei denen das bei der ConfigPageIntern eingefügte Mindestalter benötigt wird, ins Produkt JCR der Shop repository zu speichern. Die Änderungen im Code habe ich nach dem Überprüfen ins SVN committed.

Ich konnte die Kommunikation zwischen dem Model und dem JavaScript nicht hinkriegen (um die altersvariable in die browsersession zu speichern) und wandte mich nach stundenlangem recherchieren und ausprobieren zum Loris Sigrist. Er erklärte mir, dass es eine viel Simplere Methode gäbe, nämlich die Verwendung des schon existenten SessionFactory, welches ich direkt in das Model eingebunden habe. Nach einem kurzen testdurchlauf committete ich diese Änderungen auch ins SVN.

Zum Schluss exportierte ich alle neu erstellten Nodes und fügte sie in den VersionHandler des shop-moduls ein, welchen ich samt den XML Files der zu bootstrappenden Nodes ins SVN versionierte.

### Reflexion

Es ging zum grössten Teil gut voran, jedoch war das Problem wieder durch die fehlenden Informationen sowie Festsetzung auf einem nicht optimalen Lösungsansatz entstanden. Dies kann ich in Zukunft durch gründlicheres Informieren

## Montag, 13.03.2023

Tabelle Arbeitsprotokoll - Tag 6

|  |  |
| --- | --- |
| Ziele | Realisieren   1. Magnolia 2. SCSS 3. Dokumentation |
| Erreichte Ziele | Realisieren   1. Magnolia 2. Dokumentation |
| Aufgetretene Probleme | Keine |
| Beanspruchte Hilfe | Keine |
| Zeitplan | Im Zeitplan |

### Workflow

Ich hatte heute nur den halben Tag für die Probe-IPA zur Verfügung, da ich noch einen drängenden Auftrag erhielt.

Am Mittag erstellte ich die Klasse BlockUnderage.java und erstellte die BasketRules Nodes im Magnolia JCR. Ich stellte einen Codeblock des IDAgeVerificationModel.java in eine statische Funktion, damit diese auch vom BlockUnderage.java aufgerufen werden konnte, was redundanten Code verhindert hat.

Ich musste den Node im Magnolia JCR, der die IDAgeVerification Komponente ins orderFinish einbindet, eine Ebene höher, und zwar in die order page template, verschieben. Der Grund dafür ist, dass wenn das Alter noch nicht überprüft ist und folgendermassen die Basketrule auf false ist, erscheint die orderFinish Komponente erst gar nicht, weswegen die Altersüberprüfung davor erscheinen muss.

Commit neue xml’s im versionhandler, BlockUnderage, AgeVerificationModel statische funktion.

### Reflexion

Es ging eigentlich gut voran, nur die Änderung der Komponentenebene hätte ich im Voraus erkennen und planen können.

## Dienstag, 14.03.2023

Tabelle Arbeitsprotokoll - Tag 7

|  |  |
| --- | --- |
| Ziele | Realisieren   1. CSS 2. Dokumentation |
| Erreichte Ziele | Bugfixing  Realisieren   1. CSS 2. Dokumentation |
| Aufgetretene Probleme | UserDate year |
| Beanspruchte Hilfe | Keine |
| Zeitplan | Im Zeitplan |

### Workflow

Begonnen habe ich heute mit der Dokumentation des Realisierungsprozesses. Ich ergänzte Bilder und Erklärungen zu gewissen Funktionsweisen. Danach kommentierte, beziehungsweise dokumentierte den Code selbst. Dabei fiel mir auf, dass im BlockUnderage.java das eingegebene Alter des Users im Format «yyMMdd» im Falle «880101» zu 2088-01-01 anstatt 1988-01-01 konvertiert wird. Der Lösungsansatz habe ich direkt implementiert und dokumentiert.

Mir fiel auch auf, dass ein Codeblock den ich für die Aktivierung eines Nodes im Magnolia JCR in der Klasse ConfigPageIntern.java schrieb, auch ausserhalb gebraucht werden kann, weswegen ich ihn in die Klasse TemplatingFunctions.java verschob.

Anschliessend fing ich an, das CSS zu schreiben. Währenddessen suchte ich mir ein Bild aus dem Internet für das Profil-Icon des Schweizer Pass Formulars. Hierbei achtete ich auf die Nutzungsrechte und wählte eins, bei welchem man keine Quellenangabe bei Benutzung dieser angeben musste.

Schlussendlich habe ich alle bearbeiteten Dateien und das Bild ins SVN committed.

### Reflexion

Ich kam heute gut voran, doch ich merkte, dass ich an ein paar Funktionalitäten arbeitete, die nicht notwendig sind. Beispiele davon sind das Profilbild oder die Verschiebung des Aktivierungscodeblocks. Das zweite ist natürlich vorteilhaft für die Zukunft, dennoch hätte ich es mir aufschreiben und später erledigen können und mich zuerst den dringenderen offenen Tasks zuwenden.

## Mittwoch, 15.03.2023

Tabelle Arbeitsprotokoll - Tag 8

|  |  |
| --- | --- |
| Ziele | Realisieren   1. CSS 2. Dokumentation |
| Erreichte Ziele | Realisieren   1. CSS 2. Dokumentation |
| Aufgetretene Probleme | Keine |
| Beanspruchte Hilfe | Keine |
| Zeitplan | Im Zeitplan |

### Workflow

Begonnen habe ich heute damit, das CSS zu Ende zu schreiben. Das ging ziemlich gut, ich merkte jedoch, dass auf der Konto-Seite die Darstellung verschoben war, da die Elemente davor die Eigenschaft float:left; besassen. Um die Altersüberprüfungskomponente auf der Bestell- sowie Kontoübersichtsseite richtig anzeigen zu lassen musste ich die Breite auf 100% stellen und float:left; hinzufügen.

Anschliessend habe ich die Methode getMinAge() im BasketFactory erstellt und den Code aus BlockUnderage dorthin verschoben, damit ich auch aus dem IDAgeVerificationModel auf die Logik zugreifen kann. Das benötigte ich für die neu erstellte Methode needsDiplaying(), die zusätzlich zur Überprüfung ob es ein gespeichertes Alter existiert, bestimmt, ob das HTML erscheinen soll.

Anschliessend fügte ich im FTL eine [@cms.area] ein, um die Formular Komponente für die Anfrage auf Manuelle Überprüfung hinzufügen zu können. Dafür musste ich auch im Magnolia JCR diese Area der Komponente hinzufügen.

Schlussendlich committed ich alles ins SVN und schrieb die verbleibende Zeit an der Dokumentation weiter.

### Reflexion

Ich habe die Kondition des Anzeigen lassen des FTL nicht genug detailliert in der Planungsphase analysiert, weswegen ich erst jetzt die Methode needsDisplaying() eingefügt habe.

## Donnerstag, 16.03.2023

Tabelle Arbeitsprotokoll - Tag 9

|  |  |
| --- | --- |
| Ziele | Kontrollieren   1. Testen 2. Dokumentation |
| Erreichte Ziele | Kontrollieren   1. Testen 2. Dokumentation |
| Aufgetretene Probleme | Deploy Morpheus dmvini |
| Beanspruchte Hilfe | Sandro Fortunato |
| Zeitplan | Im Zeitplan |

### Workflow

Begonnen habe ich damit, die Dokumentation der Testdurchführungen vorzubereiten. Dann wollte ich die dmvini-webapp auf den «Morpheus» Tomcat7 Server (also unsere Testumgebung) deployen. Jedoch stiess ich auf einen Fehler beim Starten der Webapp. Ich wandte mich an Sandro Fortunato um Hilfe, da ich mich nicht sehr gut im Bereich Netzwerktechnik auskenne und keine Fehler begehen will, die die ganze Testumgebung beeinträchtigen könnten. Sandro leitete mich an Loris Sigrist weiter, von wem ich aber noch keine Antwort erhielt. In der Zwischenzeit informierte ich mich über die Fehlermeldung, konnte aber nicht schlau werden. Da ich schon viel Zeit damit verbracht hatte, mich mit dem Fehler auseinanderzusetzen, entschied ich mich, die Webapp auf meinem lokalen Tomcat7 Server (der Entwicklerumgebung) zu testen.

### Reflexion

Ich habe zu wenig Zeit für die Kontrollphase eingeplant. Auch ohne das heute aufgetretene Problem hätte ich mehr Zeit gebraucht, als eingeplant war.

## Freitag, 17.03.2023

Tabelle Arbeitsprotokoll - Tag 10

|  |  |
| --- | --- |
| Ziele | Auswerten   1. Auswerten 2. Dokumentation |
| Erreichte Ziele | Auswerten   1. Auswerten 2. Dokumentation |
| Aufgetretene Probleme | Keine |
| Beanspruchte Hilfe | Keine |
| Zeitplan | Im Zeitplan |

### Workflow

Heute fing ich den Tag damit an, die Dokumentation zu vervollständigen. Ich erstellte dabei ein Abkürzungsverzeichnis, das Glossar und die Bil- sowie Tabellenverzeichnisse. Anschliessend stellte ich alle Ressourcen die ich während der IPA gesammelt, entworfen oder bearbeitet hatte, in einen Ordner, welcher schlussendlich als Anhang abgegeben wird.

Zum Schluss machte ich mir Gedanken zum Gesamtprojekt und dessen Ablauf und schrieb es in der Auswertung nieder.

### Reflexion

Heute musste ich vermehrt Pausen nehmen, da die Arbeit sehr eintönig war und ich dabei sehr nervös und zappelig wurde. Abgesehen davon kam ich gut voran und konnte nach den Auszeiten mit aufgefüllter Konzentration weiterschreiben.

Kapitel 2: Projekt

# Kurzfassung des IPA-Berichtes

## Ausgangssituation

Das Ziel dieser IPA ist es, eine Altersverifikation mittels Eingabe der Kennzahlen auf dem Schweizer Pass oder der Schweizer ID umzusetzen. Diese Aufgabenstellung wurde vom Kunden Wieland AG angefragt, welche wiederum den Auftrag vom Jugendschutz erhielten. Es soll auch die Möglichkeit geben, das Alter des Endbenutzers auf Anfrage manuell im adminCentral zu setzen.

## Umsetzung

Das Projekt wurde nach der Projektmanagementmethode IPERKA umgesetzt. Diese setzt sich zusammen aus den Phasen des Informierens, Planen, Entscheiden, Realisieren, Kontrollieren und schlussendlich Auswerten.

Umgesetzt wurde der Auftrag in der Dmvini Webapp, da es keine Testinstanz der Wieland Webapp gibt. Da im shop-modul entwickelt wurde und dies in beiden Webapps einbegriffen ist, spielt dieser Unterschied keine grosse Rolle.

Das Formular sollte auf der Bestellübersichtsseite erscheinen, falls das Alter noch nicht verifiziert wurde und die Bestellung Produkte beinhaltet, die ein Mindestalter haben. Dazu muss man auf der Konto-Seite das Alter setzen können.

## Ergebnis

Alle Hauptanforderungen konnten umgesetzt werden. Die Altersüberprüfung funktioniert auf den beiden erwähnten Stellen, beim angemeldeten Enduser sowie bei Anonymous. Eine Bestellung mit alkoholhaltigen Getränken kann nicht abgeschlossen werden, ist das Alter noch nicht verifiziert oder falls es unter dem Mindestalter liegt. Das Einzige, was nicht umgesetzt werden konnte, ist die Bestätigungsmeldung nach dem Absenden des Formulars.

# Informieren

## Einleitung

Da die Kunden, für welche Inteco edv AG Webshops entwickelt, oftmals Weinhändler sind oder generell alkoholhaltige Getränke verkaufen, muss für den Jugendschutz eine Altersverifikationsmethode im Bestellprozess eingebaut sein. Damit will man versuchen, minderjährige davon abzuhalten, Alkohol zu erwerben. Da in der Schweiz das Jugendschutz und Alkoholgesetz von Kanton zu Kanton individuell geregelt wird, kann es schwierig sein, eine übergreifende und einheitliche Verifikation des Alters zu entwickeln. Meinem Verständnis nach wurde von Wieland AG mit Sitz in Thun vom Jugendschutz gefordert, auf ihrer Webseite eine Altersüberprüfung mittels Kennzahlen der Schweizer ID/Pass zu implementieren.

## Zieldefinition

* Der Abschluss der Bestellung darf erst nach der Überprüfung möglich sein.
* Die Altersüberprüfung sollte durch das Berechnen der Prüfziffern auf der Schweizer ID oder dem Schweizer Pass erfolgen. Diese müssen vom Endbenutzer in die entsprechenden Felder eines Dialoges vor Bestellende eingetragen werden.
* Bei eingeloggten Benutzern sollte das Alter nach der Verifikation im Kundenkonto in ein neues Datenfeld gespeichert werden, um bei zukünftigen Bestellungen darauf zurückgreifen können. Die Kundendaten befinden sich im Magnolia CMS.
* Bei eingeloggten Benutzern sollte auch die Möglichkeit, das Alter zu verifizieren, auf der Account-Page eingebunden sein
* Es dürfen keine weiteren Ausweisdaten gespeichert werden, ausser dem überprüften Geburtsdatum.
* Es ist zu unterscheiden zwischen Ü16 und Ü18. Manche Alkoholhaltige Getränke können laut dem Zürcher Gesetzt [(§ 25 Abs. 2 und 3, §§ 27 und 32 Abs. 2 und 3, Gastgewerbegesetz)](https://www.lexfind.ch/fe/de/tol/22460/versions/125328/de) auch Jugendliche unter 18 aber über 16 Jahren erwerben.
* Es muss eine auswählbare Option geben, diese Verifizierungsart per Prüfziffern abzuwählen und stattdessen eine manuelle Überprüfung anzufordern, falls der Endbenutzer keinen Schweizer Pass/ID besitzt oder seine Daten aus einem anderen Grund nicht eingeben möchte.
* Falls die Überprüfung durch Kennzahlen fehlschlägt oder die oben erwähnte Möglichkeit ausgewählt wird, soll eine Kontaktadresse des Shop Inhabers angezeigt werden, damit dieser die Überprüfung manuell tätigen kann.
* Damit der Kunde bei der manuellen Überprüfung das verifizierte Alter bei einem Endkunden eintragen kann, muss im Magnolia Kunden-Menu unter Shop > Users ein neues Datenfeld auf der User-Detail-Seite erfasst werden.
* Das Design der Inputs sollte intuitiv sein und klar kommunizieren, wo welche Kennzahlen eingetragen werden müssen. Ein Beispiel des Designs kann man bei [galaxus.ch](https://www.galaxus.ch/) sehen. Unten ist eine Bildschirmaufnahme dessen abgebildet.
* Diese Überprüfung sollte im magnolia-modules-shop sowie im magnolia-modules-customView implementiert werden.

## Umgebungen

Entwicklerumgebung

* localhost (Tomcat7)
  + dmvini-webapp
    - shop-ageCheck-gs2
  + firstwinenew-webapp
    - shop-div-upgr-ageCheck-gs2

Testumgebungen

* dmvini.inteco.ch (Tomcat7)
  + dmvini-webapp
    - shop-ageCheck-gs2
* firstwinenew.inteco.ch (Tomcat7)
  + firstwinenew-webapp
    - shop-div-upgr-ageCheck-gs2

## Vorhandene Ressourcen

Bildschirmaufnahme von [galaxus.ch](https://www.galaxus.ch/) (Siehe Abbildung 5)

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Abbildung Altersüberprüfung galaxus

Die mathematische Formel, um aus den Prüfziffern das Alter des Endbenutzers zu berechnen, fand ich in einem [Beitrag vom 03.06.2019 auf kleemans.ch](https://www.kleemans.ch/identit%C3%A4tskarte-berechnung-pr%C3%BCfziffer).

## Daten

Prüfziffern: <https://www.kleemans.ch/identit%C3%A4tskarte-berechnung-pr%C3%BCfziffer>

Gastgewerbegesetz: <https://www.lexfind.ch/fe/de/tol/22460/versions/125328/de>

# Planen

## Use-Case

Es wurde ein Use-Case Diagramm erstellt (Siehe Abbildung 6), sowie die Beschreibung der einzelnen Use-Cases. Das Diagramm wurde mittels Visio erstellt.

In diesem Diagramm sind die Folgenden Personen vorhanden:

* Enduser: Der Webshop-Besucher
* Kunde: Der Getränkehändler

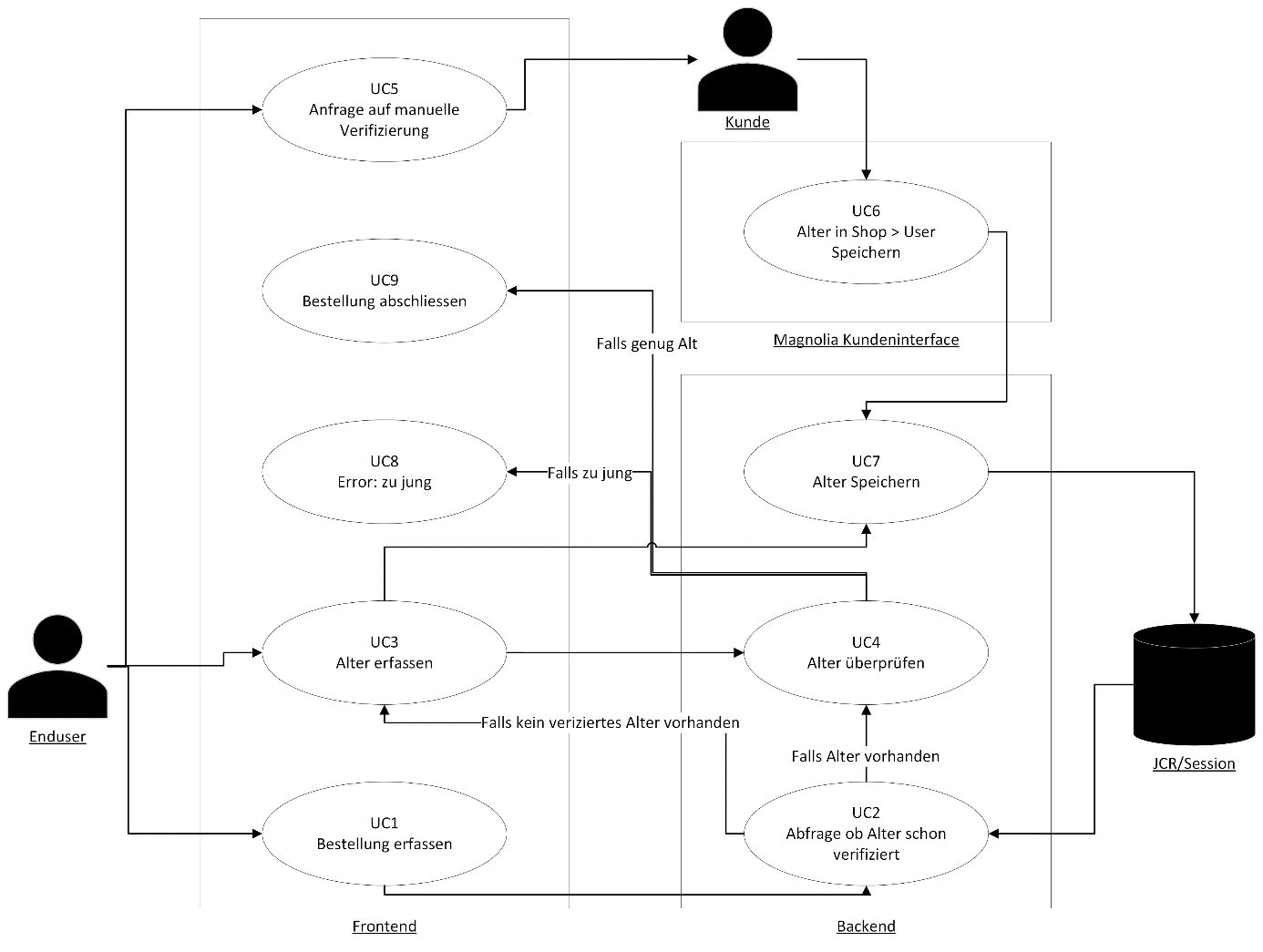


Abbildung UseCase Diagramm

Tabelle Use Case 1

|  |  |
| --- | --- |
| Nr. | UC1 |
| Name | Bestellung erfassen |
| Akteure | Enduser |
| Vorbedingung | Enduser muss sich im Webshop befinden |
| Ablauf | 1. Enduser stellt Produkte in Warenkorb 2. Enduser geht durch den Bestellprozess 3. Enduser befindet sich auf letzter Seite «Bestellung bestätigen» |
| Ergebnis | Anfrage vom verifizierten Alter an Backend (UC2) |

Tabelle Use Case 2

|  |  |
| --- | --- |
| Nr. | UC2 |
| Name | Abfrage ob Alter schon verifiziert |
| Akteure | Session / JCR |
| Vorbedingung | UC1 |
| Ablauf | * Falls User angemeldet ist: Abfrage nach Alter in den Kundendaten im JCR * Falls Anonymous: Abfrage nach Alter in der Sessionvariable |
| Ergebnis | * Falls Alter vorhanden: Alter Überprüfen (UC4) * Falls kein Alter vorhanden: Alter erfassen (UC3) |

Tabelle Use Case 3

|  |  |
| --- | --- |
| Nr. | UC3 |
| Name | Alter erfassen |
| Akteure | Enduser |
| Vorbedingung | UC2 |
| Ablauf | 1. Enduser wählt Schweizer Pass oder Schweizer ID aus 2. Enduser gibt Kennzahlen in den erschienenen input Feldern ein 3. Enduser drückt auf Knopf «Überprüfen» |
| Ergebnis | Anfrage an Backend Alter speichern und überprüfen (UC7,UC4) |

Tabelle Use Case 4

|  |  |
| --- | --- |
| Nr. | UC4 |
| Name | Alter überprüfen |
| Akteure | - |
| Vorbedingung | UC3 oder Alter schon vorhanden im JCR/Session |
| Ablauf | 1. Falls Alter noch nicht vorhanden: Aus eingegebenen Kennzahlen wird Geburtsdatum berechnet 2. True, wenn Alter des Endusers > Mindestalter für die im Warenkorb vorhandenen Produkte. Sonst False |
| Ergebnis | * False: Error (UC8) * True: Bestellung abschliessen (UC9) |

Tabelle Use Case 5

|  |  |
| --- | --- |
| Nr. | UC5 |
| Name | Anfrage auf manuelle Verifizierung |
| Akteure | Enduser, Kunde |
| Vorbedingung | UC3 |
| Ablauf | 1. Enduser wählt «manuelle Überprüfung» aus 2. Enduser kontaktiert Kunde durch die ihm angezeigten Optionen |
| Ergebnis | Kunde verifiziert Alter und speichert es (UC6) |

Tabelle Use Case 6

|  |  |
| --- | --- |
| Nr. | UC6 |
| Name | Alter in Shop > User Speichern |
| Akteure | Kunde |
| Vorbedingung | UC5 |
| Ablauf | 1. Kunde meldet sich im Magnolia an und wählt Shop>User aus 2. Kunde wählt den zu bearbeitenden User aus 3. Kunde setzt Datenfeld «verifiziertes Alter» auf ihm bekanntes Alter |
| Ergebnis | Anfrage an Backend Alter Speichern (UC7) |

Tabelle Use Case 7

|  |  |
| --- | --- |
| Nr. | UC7 |
| Name | Alter Speichern |
| Akteure | JCR / Session |
| Vorbedingung | UC3 oder UC6 |
| Ablauf | * Falls UC3 und Enduser ist Anonymous: Alter in Session speichern * Sonst Alter in JCR beim dazugehörigen User speichern |
| Ergebnis | Alter wurde gespeichert |

Tabelle Use Case 8

|  |  |
| --- | --- |
| Nr. | UC8 |
| Name | Error: zu jung |
| Akteure | Enduser |
| Vorbedingung | UC4 |
| Ablauf | Eine Meldung wird angezeigt, dass der Enduser zu jung ist, um zu bestellen |
| Ergebnis | Enduser kann die Bestellung nicht abschliessen |

Tabelle Use Case 9

|  |  |
| --- | --- |
| Nr. | UC9 |
| Name | Bestellung abschliessen |
| Akteure | Enduser |
| Vorbedingung | UC4 |
| Ablauf | Eine Meldung wird angezeigt, dass der Enduser zu genug alt ist, um zu bestellen |
| Ergebnis | Enduser kann die Bestellung abschliessen |

## Flussdiagramm

Um den Ablauf des Programms zu verdeutlichen, erstellte ich ein Flussdiagramm (Siehe Abbildung 7). Dieses wurde mittels draw.io erstellt.

Ein Bild, das Diagramm enthält.

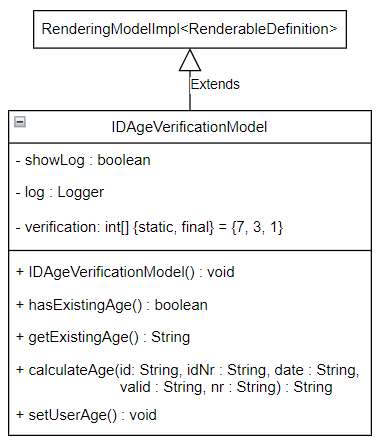
Automatisch generierte Beschreibung

Abbildung Flussdiagramm

## Mockup

Ich habe mich dafür entschieden, kein Mockup zu erstellen, da ich als Referenz die Bildschirmaufnahme von [galaxus.ch](https://www.galaxus.ch/) nutzen werde. [(Siehe Abbildung 5)](#_Vorhandene_Ressourcen)

## Klassendiagramm

Ich habe für das IDAgeVerificationModel.java mittels draw.io ein Klassendiagramm erstellt (Siehe Abbildung 6).

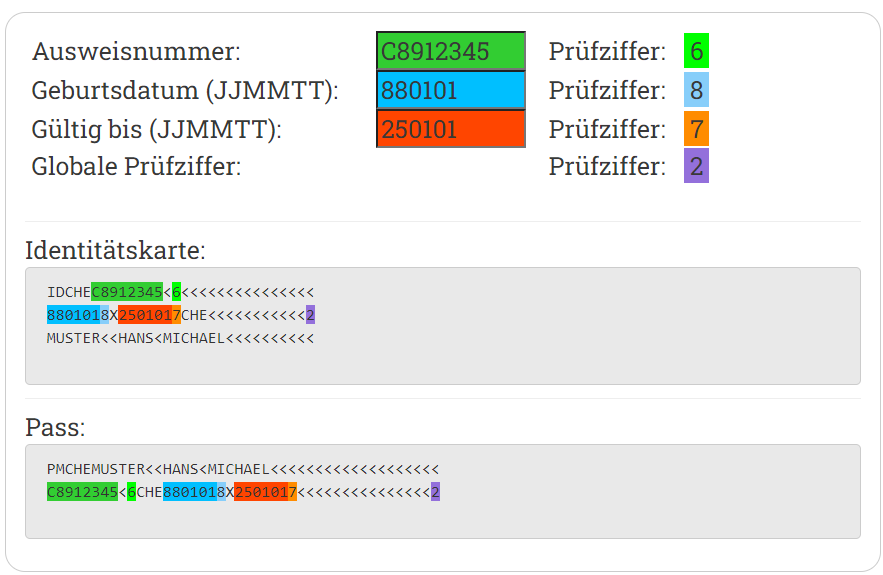
Unten habe ich einen Screenshot aus dem [Beitrag der Seite kleemans](https://www.kleemans.ch/identit%C3%A4tskarte-berechnung-pr%C3%BCfziffer) eingefügt (Siehe Abbildung 7). Von dort aus habe ich die Parameter der Methode calculateAge() erarbeitet.

Abbildung Klassendiagramm IDAgeVerificationModel

Abbildung Kleemans Kennzahlen

## Testkonzept

### Testumgebung

Die Applikation werde ich mit folgenden Browsern testen. Ich habe sie ausgewählt, da sie alle verschiedene «Rendering Engines» brauchen und die weitverbreitetsten Browser sind

Tabelle Testbrowser

|  |  |  |
| --- | --- | --- |
| Nr | Browser | Version |
| B1 | Chrome | Version 112.0.5596.2 (Offizieller Build) dev (64-Bit) |
| B2 | Firefox | Version 111.0b7 (64-bit) |
| B3 | Edge | Version 110.0.1587.57 (Official build) (64-bit) |
| B4 | Safari |  |

Ich werde die Funktionalität zuerst lokal bei meiner dmvini-webapp und anschliessend auf unserer Testumgebung «Morpheus» überprüfen, um sicherzugehen, dass die Webapp auf der Live-Umgebung wie geplant funktionieren wird.

Tabelle Testumgebungen

|  |  |
| --- | --- |
| Nr | Umgebung |
| U1 | Tomcat7 localhost |
| U2 | dmvini.inteco.ch (Tomcat7 «Morpheus») |

### Testfälle

Der Testfall für UC1 wurde nicht aufgeschrieben, da das Ergebnis des Einbindens der ftl Datei IDAgeVerificationComp.ftl klar ersichtlich ist und ich die Funktionsweise dessen Stärker gewichte.

Tabelle Testfall 1

|  |  |
| --- | --- |
| Nr | 1 |
| Name | Bestellvorgang verifizierter User angemeldet |
| Use-Case | UC1 |
| Akteur | Enduser |
| Vorbedingung | * Enduser befindet sich auf Seite «Bestellung abschliessen» * Bestellung beinhaltet Produkte, bei denen Alter überprüft werden muss * Enduser ist eingeloggt * Das verifizierte Alter des Users ist im JCR gespeichert |
| Testablauf | * Als User mit verifiziertem Alter einloggen * Produkte in den Warenkorb mit Mindestalter > 0 tun * Auf Bestellübersichtsseite gehen |
| Erwartetes Ergebnis | Resultat Testfälle Nr 11, 12 |

Tabelle Testfall 2

|  |  |
| --- | --- |
| Nr | 2 |
| Name | Bestellvorgang nicht verifizierter User angemeldet |
| Use-Case | UC1 |
| Akteur | Enduser |
| Vorbedingung | * Enduser befindet sich auf Seite «Bestellung abschliessen» * Bestellung beinhaltet Produkte, bei denen Alter überprüft werden muss * Enduser ist eingeloggt * Es ist kein verifiziertes Alter des Users im JCR gespeichert |
| Testablauf | * Als User ohne verifiziertem Alter einloggen * Produkte in den Warenkorb mit Mindestalter > 0 tun * Auf Bestellübersichtsseite gehen |
| Erwartetes Ergebnis | Auf der Seite erscheinen die Optionen, das Alter zu verifizieren.  Der Bestellprozess kann noch nicht weitergeführt werden und es erscheint die Fehlermeldung «Alter noch nicht verifiziert» |

Der Testfall für den Fall, dass der User angemeldet ist und kein Alter im JCR aber ein Alter in der Session gespeichert ist, wurde nicht aufgelistet, da die zwei Prozesse der Altersüberprüfung eines Eingeloggten Users und Anonymous getrennt sind und sich nicht gegenseitig «stören» können. In diesem gerade beschriebenen Fall wird der User trotzdem nach der Verifikation des Alters gefragt, auch wenn schon ein Verifiziertes Alter in der Session existiert.

Tabelle Testfall 3

|  |  |
| --- | --- |
| Nr | 3 |
| Name | Bestellvorgang verifizierter User anonymous |
| Use-Case | UC1 |
| Akteur | Enduser |
| Vorbedingung | * Enduser befindet sich auf Seite «Bestellung abschliessen» * Bestellung beinhaltet Produkte, bei denen Alter überprüft werden muss * Enduser ist nicht eingeloggt * Das verifizierte Alter ist in der Session gespeichert |
| Testablauf | * Produkte in den Warenkorb mit Mindestalter > 0 tun * Auf Bestellübersichtsseite gehen * Alter verifizieren |
| Erwartetes Ergebnis | Testfälle Nr 11, 12 |

Tabelle Testfall 4

|  |  |
| --- | --- |
| Nr | 4 |
| Name | Bestellvorgang nicht verifizierter User anonymous |
| Use-Case | UC1 |
| Akteur | Enduser |
| Vorbedingung | * Enduser befindet sich auf Seite «Bestellung abschliessen» * Bestellung beinhaltet Produkte, bei denen Alter überprüft werden muss * Enduser ist nicht eingeloggt * Es ist kein verifiziertes Alter in der Session gespeichert |
| Testablauf | * Produkte in den Warenkorb mit Mindestalter > 0 tun * Auf Bestellübersichtsseite gehen |
| Erwartetes Ergebnis | Auf der Seite erscheinen die Optionen, das Alter zu verifizieren.  Der Bestellprozess kann noch nicht weitergeführt werden und es erscheint die Fehlermeldung «Alter noch nicht verifiziert» |

Tabelle Testfall 5

|  |  |
| --- | --- |
| Nr | 5 |
| Name | Anonymous Verifikationsvorgang ID korrekt |
| Use-Case | UC3 |
| Akteur | Enduser |
| Vorbedingung | * Enduser befindet sich auf Seite «Bestellung abschliessen» * Enduser ist nicht eingeloggt * Bestellung beinhaltet Produkte, bei denen Alter überprüft werden muss * Das Alter des Endusers ist noch nicht verifiziert |
| Testablauf | * Produkte in den Warenkorb mit Mindestalter > 0 tun * Auf Bestellübersichtsseite gehen * Alter verifizieren durch ID |
| Erwartetes Ergebnis | Auf der Seite erscheint eine Konfirmationsmeldung und die Altersüberprüfungkomponente verschwindet.  Das Alter wird in der Session gespeichert  Resultat Testfälle Nr 11, 12 |

Tabelle Testfall 6

|  |  |
| --- | --- |
| Nr | 6 |
| Name | Anonymous Verifikationsvorgang Pass korrekt |
| Use-Case | UC3 |
| Akteur | Enduser |
| Vorbedingung | * Enduser befindet sich auf Seite «Bestellung abschliessen» * Enduser ist nicht eingeloggt * Bestellung beinhaltet Produkte, bei denen Alter überprüft werden muss * Das Alter des Endusers ist noch nicht verifiziert |
| Testablauf | * Produkte in den Warenkorb mit Mindestalter > 0 tun * Auf Bestellübersichtsseite gehen * Alter verifizieren durch Pass |
| Erwartetes Ergebnis | Auf der Seite erscheint eine Konfirmationsmeldung und die Altersüberprüfungkomponente verschwindet.  Das Alter wird in der Session gespeichert  Resultat Testfälle Nr 11, 12 |

Tabelle Testfall 7

|  |  |
| --- | --- |
| Nr | 7 |
| Name | Verifikationsvorgang eingeloggter User ID korrekt |
| Use-Case | UC3 |
| Akteur | Enduser |
| Vorbedingung | * Enduser befindet sich auf Seite «Bestellung abschliessen» * Enduser ist eingeloggt * Bestellung beinhaltet Produkte, bei denen Alter überprüft werden muss * Das Alter des Endusers ist noch nicht verifiziert |
| Testablauf | * Als User ohne verifiziertem Alter einloggen * Produkte in den Warenkorb mit Mindestalter > 0 tun * Auf Bestellübersichtsseite gehen * Alter verifizieren durch ID |
| Erwartetes Ergebnis | Auf der Seite erscheint eine Konfirmationsmeldung und die Altersüberprüfungkomponente verschwindet.  Das Alter wird im JCR gespeichert  Resultat Testfälle Nr 11, 12 |

Tabelle Testfall 8

|  |  |
| --- | --- |
| Nr | 8 |
| Name | Verifikationsvorgang eingeloggter User Pass korrekt |
| Use-Case | UC3 |
| Akteur | Enduser |
| Vorbedingung | * Enduser befindet sich auf Seite «Bestellung abschliessen» * Enduser ist eingeloggt * Bestellung beinhaltet Produkte, bei denen Alter überprüft werden muss * Das Alter des Endusers ist noch nicht verifiziert |
| Testablauf | * Als User ohne verifiziertem Alter einloggen * Produkte in den Warenkorb mit Mindestalter > 0 tun * Auf Bestellübersichtsseite gehen * Alter verifizieren durch Pass |
| Erwartetes Ergebnis | Auf der Seite erscheint eine Konfirmationsmeldung und die Altersüberprüfungkomponente verschwindet.  Das Alter wird im JCR gespeichert  Resultat Testfälle Nr 11, 12 |

Tabelle Testfall 9

|  |  |
| --- | --- |
| Nr | 9 |
| Name | Verifikationsvorgang nicht korrekt |
| Use-Case | UC3 |
| Akteur | Enduser |
| Vorbedingung | * Enduser befindet sich auf Seite «Bestellung abschliessen» * Bestellung beinhaltet Produkte, bei denen Alter überprüft werden muss * Das Alter des Endusers ist noch nicht verifiziert |
| Testablauf | * Als User ohne verifiziertem Alter einloggen * Produkte in den Warenkorb mit Mindestalter > 0 tun * Auf Bestellübersichtsseite gehen * Alter verifizieren lassen mit fehlerhafter Eingabe |
| Erwartetes Ergebnis | Auf der Seite erscheint die Fehlermeldung «Eingabe nicht korrekt». |

Der Testfall «Verifikationsvorgang nicht korrekt» wurde nicht aufgeteilt (Anonymous und eingeloggter User, ID und Pass) weil der Vorgang und das Resultat bei allen gleich ist.

Tabelle Testfall 10

|  |  |
| --- | --- |
| Nr | 10 |
| Name | Bestellung ohne Altersverifikation |
| Use-Case | UC8, UC9 |
| Akteur | Enduser |
| Vorbedingung | * Enduser befindet sich auf Seite «Bestellung abschliessen» * Bestellung beinhaltet keine Produkte, bei denen das Alter überprüft werden muss * Das Alter des Endusers ist noch nicht verifiziert |
| Testablauf | * Als User ohne verifiziertem Alter einloggen * Produkte in den Warenkorb mit Mindestalter 0 tun * Auf Bestellübersichtsseite gehen |
| Erwartetes Ergebnis | Der Bestellprozess kann normal weitergeführt werden. |

Die Testfälle zur Überprüfung, ob der Enduser mit verifiziertem Alter eine Bestellung abschliessen kann, die keine Altersverifikation erfordert, habe ich ausgelassen, da es voraussichtlich die gleichen Resultate wie Testfall Nr. 7 beinhalten wird.

Tabelle Testfall 11

|  |  |
| --- | --- |
| Nr | 11 |
| Name | Bestellung mit Altersverifikation Ü16 |
| Use-Case | UC8, UC9 |
| Akteur | Enduser |
| Beschreibung |  |
| Vorbedingung | * Enduser befindet sich auf Seite «Bestellung abschliessen» * Bestellung beinhaltet Produkte, bei denen das Mindestalter 16+ beträgt |
| Testablauf | * Produkte in den Warenkorb mit Mindestalter 16 tun * Auf Bestellübersichtsseite gehen |
| Erwartetes Ergebnis | Falls Alter über 16: Der Bestellprozess kann normal weitergeführt werden.  Sonst: Fehlermeldung «zu jung» und Bestellvorgang kann nicht abgeschlossen werden. |

Tabelle Testfall 12

|  |  |
| --- | --- |
| Nr | 12 |
| Name | Bestellung mit Altersverifikation Ü18 |
| Use-Case | UC8, UC9 |
| Akteur | Enduser |
| Beschreibung |  |
| Vorbedingung | * Enduser befindet sich auf Seite «Bestellung abschliessen» * Bestellung beinhaltet Produkte, bei denen das Mindestalter 18+ beträgt |
| Testablauf | * Als User ohne verifiziertem Alter einloggen * Produkte in den Warenkorb mit Mindestalter 18 tun * Auf Bestellübersichtsseite gehen |
| Erwartetes Ergebnis | Falls Alter über 18: Der Bestellprozess kann normal weitergeführt werden.  Sonst: Fehlermeldung «zu jung» und Bestellvorgang kann nicht abgeschlossen werden. |

Tabelle Testfall 13

|  |  |
| --- | --- |
| Nr | 13 |
| Name | Alter setzen im Magnolia Shop > Users |
| Use-Case | UC6 |
| Akteur | Kunde |
| Beschreibung |  |
| Vorbedingung | * Kunde ist im Magnolia angemeldet * Kunde befindet sich auf Seite Shop > User |
| Testablauf | * Mit Kundenkonto im Magnolia adminCentral einloggen * Auf Shop > User gehen * Bei einem User das Verifizierte Alter setzen |
| Erwartetes Ergebnis | Das eingegebene Alter ist im User JCR gespeichert und wird im dazugehörigen Feld auf der Seite angezeigt. |

Tabelle Testfall 14

|  |  |
| --- | --- |
| Nr | 14 |
| Name | Formularversendung manuelle Überprüfung |
| Use-Case | UC5 |
| Akteur | Enduser |
| Beschreibung |  |
| Vorbedingung | * Enduser befindet sich auf der Seite mit dem Formular |
| Testablauf | Formular Ausfüllen mit gültiger E-Mail Adresse und absenden. |
| Erwartetes Ergebnis | Email wird an Endbenutzer und an Kunden versendet. |

Tabelle Testfall 15

|  |  |
| --- | --- |
| Nr | 15 |
| Name | Anzeige Formular Bestellübersicht |
| Use-Case | UC3 |
| Akteur | Enduser |
| Beschreibung |  |
| Vorbedingung | * Enduser befindet sich auf der Bestellübersichtsseite * Bestellung enthält Produkte mit Mindestalter über 0 * Alter des Endusers ist noch nicht verifiziert |
| Testablauf | * Als User ohne verifiziertem Alter einloggen * Produkte in den Warenkorb mit Mindestalter > 0 tun * Auf Bestellübersichtsseite gehen |
| Erwartetes Ergebnis | Formular wird korrekt angezeigt:   * Breite 100% * ID und Pass Formular ähneln der schweizer ID/Pass * Die Input Felder sind klar ersichtlich und intuitiv |

Tabelle Testfall 16

|  |  |
| --- | --- |
| Nr | 16 |
| Name | Anzeige Formular Konto |
| Use-Case | UC3 |
| Akteur | Enduser |
| Beschreibung |  |
| Vorbedingung | * Enduser befindet sich auf der Kontoseite * Alter des Endusers ist noch nicht verifiziert |
| Testablauf | * Als User ohne verifiziertem Alter einloggen * Produkte in den Warenkorb mit Mindestalter > 0 tu * Auf Kontoseite gehen |
| Erwartetes Ergebnis | Formular wird korrekt angezeigt:   * Breite 100% * ID und Pass Formular ähneln der schweizer ID/Pass * Die Input Felder sind klar ersichtlich und intuitiv |

## Weiterer Prozess

Um die Realisierungsphase zu ordnen und vereinfachen, habe ich mir den folgenden Vorgehensplan erstellt:

1. IDAgeVerificationComp.ftl erstellen
2. IDAgeverificationComp.scss erstellen
3. IDAgeVerificationModel.java erstellen
4. BlockUnderage.java erstellen
5. ConfigPageIntern.html und ConfigPageIntern.java erstellen
6. Im Magnolia IDAgeVerificationComp Node erstellen und ftl sowie java Dateien verlinken
7. Im Magnolia die Verlinkung in der Orders Page erstellen, damit die Komponente dort eingefügt werden kann
8. Im Magnolia einen Node erstellen mit den Mindestalter Properties zu den verschiedenen Getränke-Kategorien (Eventuell Importer schreiben, damit die Kategorien importiert werden)
9. Freemarker Code schreiben
10. Im messages\_de.properties Bezeichnungen einfügen, i18n implementieren
11. In den französischen, englischen und italienischen .properties Dateien Bezeichnungen Übersetzen
12. it und fr. properties Dateien zum Überprüfen abgeben, da meine Italienisch- und Französischkenntnisse unzureichend sind
13. JS/JQuery Code schreiben
14. Java Code schreiben
15. CSS/SCSS Code schreiben
16. In der SettingsShopPage.html die Felder zur Erfassung der Werte des Punkts 6.
17. In der SettingsShopPage.java die zugehörige Methode hinzufügen
18. Ins CustomView übertragen

# Entscheiden

## Mindestalter Properties

Als Voraussetzung für die Realisierungsphase muss ich mich noch bei Punkt 6. [des Plans zur Entwicklung (K.2.3.7)](#_Weiterer_Prozess) entscheiden, welcher Pfad dafür am sinnvollsten ist.

Die zwei Optionen dafür wären wie folgt:

### Variante 1

JCR Pfad: shop:/categoriesMinAge

Vorteile:

* In der shop Repository sind die meisten Datensätze gespeichert, die mit dem Shop und den Produkten gespeichert sind. Meiner Meinung nach gehört das Mindestalter intuitiv dort hin.

Nachteile:

* Es wäre umständlicher, mit dem bestehenden Backend der Einstellungen > Shop Page im Magnolia die Werte in der shop Repository umzuändern, da die vorhandenen Methoden auf die config Repository konfiguriert sind.

### Variante 2

JCR Pfad: config:/modules/shop/settings/categoriesMinAge

Vorteile:

* In diesem Pfad sind Konfigurationen des Shops gespeichert, was den Ort des Mindestalters auch nicht komplett verkehrt macht.
* Es können durch einen einfachen Prozess die Eingabefelder des Mindestalters der Kategories in die Konfigurationsseite im Magnolia eingefügt werden, ohne dass die Logik dahinter neu geschrieben oder abgeändert werden muss.

Nachteile:

* Falls der Import der Kategorien implementiert werden sollte, ist der Pfad weniger intuitiv, da die meisten Imports Daten ins shop JCR speichern.

### Entscheid

Entschieden habe ich mich für Variante 2, da der Aufwand kleiner ist und ich die Beibehaltung des schon funktionierenden Backends stärker gewichte als einen «intuitiveren» Speicherort.

## Kategorienimport

Ich hatte auch die Idee, einen Importer zu schreiben, oder den bestehenden Produktimporter so anzupassen, dass die Kategorien ins JCR geschrieben werden (natürlich ohne das Alter zu überschreiben falls schon hinterlegt), da nicht jeder Kunde die gleichen Kategorien hat. Da das aber zusätzlichen Zeitaufwand bedeutet und kein Kernfeature des Auftrags ist, werde ich es nur erledigen, sobald alles andere zuerst fertig entwickelt ist und noch Zeit dafür bleibt

# Realisieren

## Überprüfung durch Kennzahlen

### Freemarker

Als erstes erstellte ich im Ordner account eine Freemarker Datei namens IDAgeVerification.ftl in welcher sich das Formular und verschiedene Abfragen befinden werden. Zu Beginn schrieb ich nur das Grundgerüst der Input-Felder und einen <select>, mit welchem man zwischen den verschiedenen Optionen (ID/Pass/Manuell) wählen kann.

Anschliessend habe ich auf eine Abfrage, ob ein Verifiziertes Datum schon hinterlegt ist, das Formular ein- respektive ausgeblendet. Da das Formular im Bestellverlauf nicht angezeigt werden soll, falls die Produkte kein Mindestalter haben, erstellte ich im Model eine Funktion namens needsDisplaying(), die in der Freemarker Komponente aufgerufen wird und die Anzeige des Formulars ein- oder ausblendet.

Zusätzlich zum selbsterklärenden Design des Formulars wurden die Input Felder wurden mit einem placeholder versehen, damit dem User klar ist, wie die Zahlen eingegeben werden müssen. Als Sicherheitsvorkehrung schrieb ich für jedes Feld eine Regular Expression, mit welcher nur valide Eingaben akzeptiert werden.

<input type="text" size="8" maxlength="8" minlength="8" placeholder="C4328590" id="pass-idInput" name="id" pattern="[a-zA-Z]{1}[0-9]{7}" required>

### I18n

Gleich nachdem die Struktur beendet war, erstellte ich im messages\_de.properties die benötigten Werte, damit die Mehrsprachigkeit mit i18n schon implementiert ist. Die weiteren Sprachen (en/fr/it) werde ich im späteren Verlauf in die dazugehörigen Files hinzufügen.

### JavaScript

Um bei der Auswahl einer <option> aus dem oben erwähnten <select> Element das entsprechende Formular anzuzeigen, schrieb ich eine Methode im IDAgeVerification.js, welche aufgerufen wird, sobald eine Option ausgewählt wird.

#### IDAgeVerification.js

// Displays selected age Verification Method

function selectAgeVerification(select){

// Get the selected option

var selectedOptionValue = $(select).val();

// Add the 'hidden' class to all forms that don't match the selected option value

$('.age-verification > div').not('.' + selectedOptionValue).addClass('hidden');

// Remove the 'hidden' class from the form that matches the selected option value

$('.age-verification > div.' + selectedOptionValue).removeClass('hidden');

}

#### IDAgeVerification.ftl

<label for="verification-options">${i18n['IDAgeVerification.choose']}</label>

<select name="verification-options" id="verification-options" onchange="selectAgeVerification(this);">

    <option value="id-verification" selected>${i18n['IDAgeVerification.swissID']}</option>

    <option value="pass-verification">${i18n['IDAgeVerification.swissPass']}</option>

    <option value="other-verification">${i18n['IDAgeVerification.other']}</option>

</select>

### CSS

Wie oben erwähnt, werden die Elemente mit der Klasse hidden nicht angezeigt.

.hidden {

display:none;

}

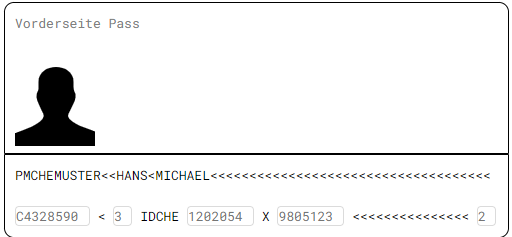
Ich habe eine Monospace Schriftart gewählt und von [google fonts](https://fonts.google.com/) die .ttf Dateien für [Roboto Mono](https://fonts.google.com/specimen/Roboto+Mono?category=Monospace) heruntergeladen. Das tat ich, damit die Zeichen gleichmässig verteilt sind und damit sie denen auf der ID ähneln (Siehe Abbildung 10). Das RobotoMono-Regular.ttf wurde im Ordner webfonts gespeichert und in webfonts.css eingebunden. Dazu habe ich das Interface des Formulars nach dem Aussehen der ID, respektive des Passes konzipiert.

Abbildung Design Formular

Für das Profilbild im Element für den Schweizer Pass habe ich ein SVG von [uxwing](https://uxwing.com/), frei für kommerziellen gebrauch, heruntergeladen und im FTL eingefügt. (Siehe Abbildung 8)

### Java

Das Backend dieser Komponente wurde mit Java implementiert.

#### IDAgeVerificationModel.java

Im Model ist die Logik der Überprüfung vorhanden. Sobald diese Klasse instanziiert wird,

Ich habe die statische Methode getVerifiedUserDate() erstellt, um auf diese aus anderen Klassen (spezifisch dem BlockUnderage.java) zugreifen zu können.

#### SessionSettingsFactory.java

In der SessionSettingsFactory.java sind Methoden für die Manipulation der Attribute der Magnolia HttpSession definiert. Diese Klasse war bereits vorhanden, es mussten jedoch die get() und set() Methoden für die Variable verifiedDate hinzugefügt werden.

### Magnolia

Um die Verbindung zwischen dem FTL und des Models herzustellen sowie ermöglichen, dass die Komponente im Bearbeitungsmodus der Webseite eingefügt werden kann, müssen einige Nodes im JCR erstellt werden.

#### Konfiguration

Dies ist der Konfigurations-Node der IDAgeVerification Komponente, welcher auf diesem Pfad im JCR liegt:

config:/modules/shop/templates/components/IDAgeVerification

Er enthält folgende Properties:

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Abbildung IDAgeVerification Konfigurationsnode

Die Eigenschaften title und description werden, wie die im ftl mit i18n eingefügten Werte, aus dem im i18nBasename festgelegten .properties File gelesen. Weiter wird eine Verbindung zum FTL-View und Java Model in den Eigenschaften templateScript respektive modelClass erstellt.

#### Komponente

Dies sind die Einbindungen der Komponente im order sowie account, welche schlussendlich auf der Bestellübersichtseite erscheinen, um bei der Bestellung das Alter zu bestätigen oder auf der Seite Konto, um das Alter direkt zu setzen.

Diese Verlinkungen liegen auf diesen beiden Orten im JCR:

config:/modules/shop/templates/pages/order/areas/content/availableComponents/IDAgeVerification

config:/modules/shop/templates/pages/account/areas/content/availableComponents/IDAgeVerification

Ausgegraut wurden die bestehenden Knotenpunkte. Der Node unter account wird nicht abgebildet, da er die gleiche property enthält wie die anderen, nur unter einem anderen Pfad liegt.

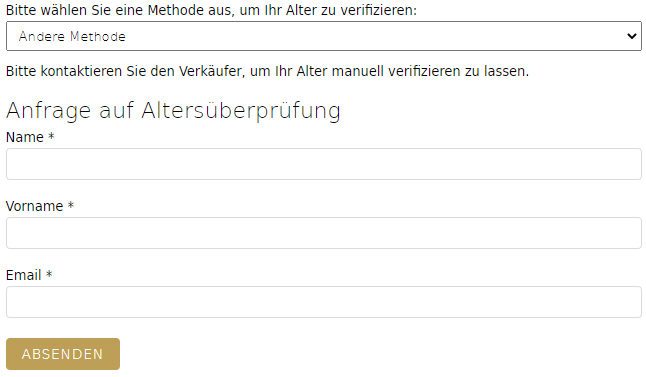
Ein Bild, das Text, Tisch enthält.

Automatisch generierte Beschreibung

Abbildung IDAgeVerification Komponentennode

## Manuelle Überprüfung

### Komponente

Bei der Option der manuellen Überprüfung wollte ich ein Formular einbinden, welches beim submit dem User ein Konfirmationsmail und dem Kunden ein Informationsmail mit den Angaben des Users versendet. Da schon Templates für Formulare entwickelt wurden, nutzte ich diese und baute ein Formular mit den Feldern «E-Mail» «Name» und «Vorname» zusammen.

Um das Formular innerhalb der Komponente einfügen zu können, musste ich im IDAgeVerification.ftl den folgenden Code-Snippet einfügen:

Abbildung Anfrageformular

[@cms.area *name="contactForm" /*]

Ein Bild, das Text enthält.

Automatisch generierte BeschreibungIch habe die «FormEmail» Komponente gewählt, da diese schon zwei Templates für die Konfiguration eines Emails enthält. Für Testzwecke habe ich in der Absender- und Empfänger-Email Adresse meine, beziehungsweise die inteco webmaster Email angegeben. Diese werden in der Produktion auf die des Kunden abgeändert. Die Komponente habe ich wie folgt konfiguriert:

Abbildung FormEmail Dialog

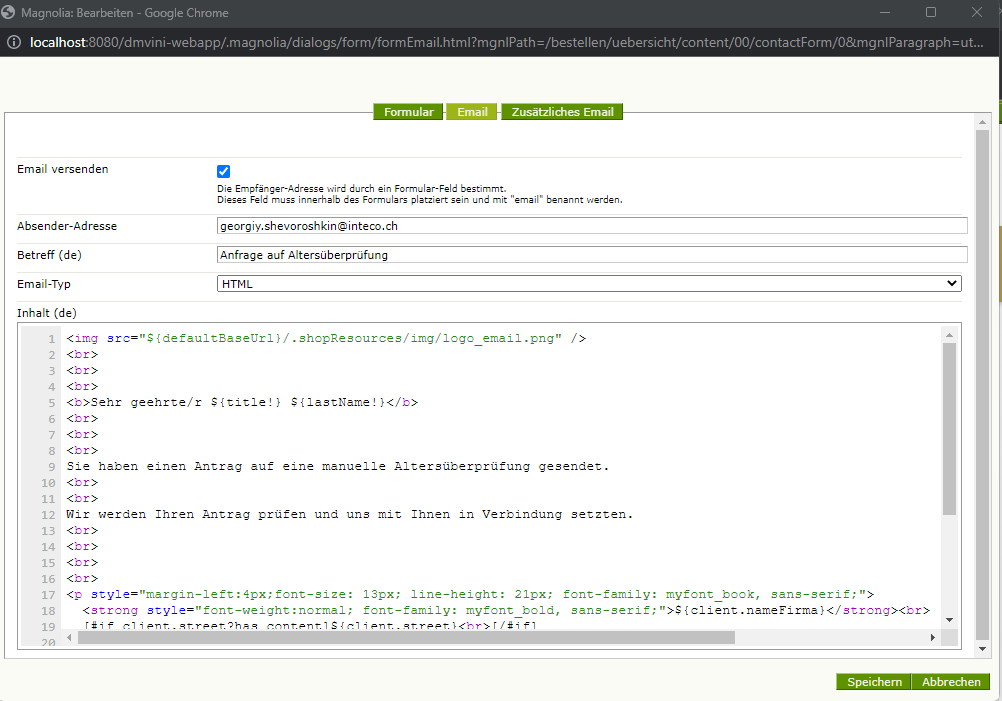


Abbildung FormEmail Dialog Email1

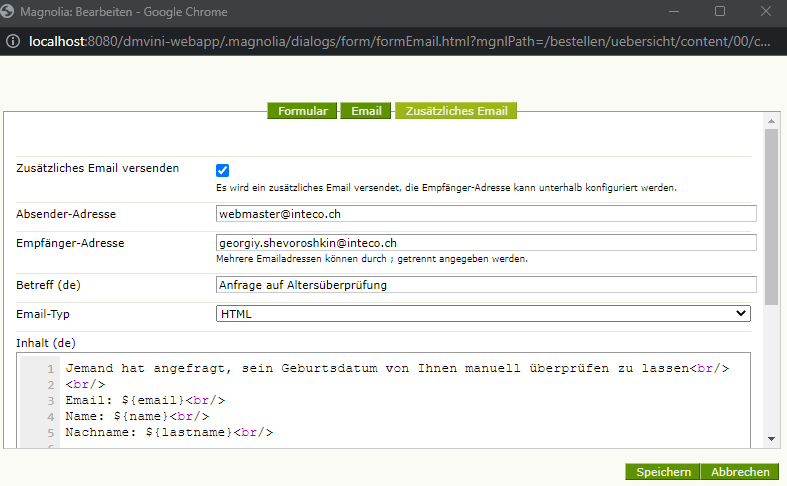


Abbildung FormEmail Dialog Email2

## Config Page Intern

Um das Mindestalter der Kategorien zu definieren, erstellte ich die «Config Page intern» im Magnolia adminCentral, auf welche nur die Entwickler Zugriff haben sollen. (Siehe Abbildung 20) Dafür wurden die Dateien ConfigPageIntern.html für das Interface und ConfigPageIntern.java für die dazugehörige Logik, nämlich das Speichern der Werte im JCR (Siehe Abbildung 17) entwickelt.

### HTML

Ich kopierte das Grundgerüst aus dem schon existenten ConfigPageIntern.html aus dem CustomView modul. Dann erstellte ich ein Formular mit den Input Feldern für das Mindestalter der gegebenen Kategorien. Die Kategorien wurden zunächst hardcoded, doch das Ziel ist es, sie schlussendlich mithilfe eines Importers dynamisch zu erstellen.

Abbildung Mindestalter Inputs

### Java

Das Backend dieser Seite wurde im ConfigageIntern.java erstellt. Im TemplatingFunctions.java wurde dabei auch eine Methode hinzugefügt.

#### ConfigPageIntern

In der Java Klasse wurden die get() Methoden für die Werte der einzelnen Kategorien implementiert, damit diese schon als Value im Input Element erscheinen, falls sie im JCR schon existieren. Dazu habe ich die saveMinAge() Methode geschrieben, welche beim Absenden des Formulars ausgeführt wird und die eingegebenen Werte aus dem Request holt und ins JCR speichert. Dazu aktiviert es den Node auf die www-Instanz mithilfe der neu erstellten Funktion TemplatingFunctions.activateNode().

#### TemplatingFunctions

Da der Aktivierungsprozess in mehreren Bereichen der Webapp gebraucht werden kann und es noch keine einheitliche Methode gab, um dies zu bezwecken, erstellte ich in der Klasse TemplatingFunctions.java die Methode activateNode(String workspace, String path), welche den Node auf dem der mit den Parametern mitgegebenen Pfad liegt.

### Rubrik

Der erste Node musste erstellt werden, um im Menu (links) die Auswahlmöglichkeit «Intern» unter der Rubrik «Einstellungen» einzufügen und die neu entwickelte Seite verlinken (Siehe Abbildung 18). Dieser befindet sich auf dem folgenden Pfad:

config:/modules/adminInterface/config/menu/settings/configPageIntern

Ausgegraut wurden die bestehenden Knotenpunkte.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Abbildung ConfiPageIntern Rubriknode

### Magnolia

Weiter musste das dazugehörige Backend mithilfe der Property class im Magnolia-Konfigurationsnode der Seite zugewiesen werden. Der Node mit der erklärten Eigenschaft befindet sich auf diesem Pfad:

config:/modules/shop/pages/ConfigPageIntern



Abbildung ConfigPageIntern Konfigurationsnode

### Berechtigungen

Die Entwickler (Rolle superuser) und der Kunde (Rolle admin) können auf das Magnolia adminCentral zugreifen. Um zu gewährleisten, dass nur die Entwickler Zugriff auf die Einstellung des Mindestalters haben, musste ich die Berechtigungen der Rolle admin wie folgt anpassen: Die unterste Zeile wurde eingefügt. (Siehe Abbildung 20)

Ein Bild, das Tisch enthält.

Automatisch generierte Beschreibung

Abbildung ConfigPageIntern Berechtigungen

### JCR

Die eingegebenen Werte werden am folgenden Ort im JCR gespeichert:

config:modules/shop/settings/minAge

Abbildung minAge JCR

## Produktimport

Da es im Standartshop Modul (mit der DMVini Webapp und Datenbank) keine einheitliche Kategorienstruktur gibt, musste ich einen neuen Produktproperty-Importer Node mit folgender MySQL Abfrageerstellen. Dieser erstellt im Produkt-Node des Shop-JCR ein Property ageCategory (Siehe Abbildung 23), welches die vier möglichen Kategorien enthält. Da Biere in der Datenbank unter der Oberkategorie «Diverse Getränke» liegen, musste ich auf den Statistikcode stcd abfragen, ansonsten konnte ich die Werte aus dem os20\_typ Auslesen.

### MySQL

Dieses MySQL Statement wurde benutzt, um den Wert zu bestimmen, der in das Produkt-JCR Importiert wird:

SELECT CASE

WHEN os20\_typ IN ('rotw' , 'weis', 'sues', 'rose', 'scha') THEN 'weine'

WHEN os20\_typ = 'spir' THEN 'spirituosen'

WHEN LEFT(os20\_shop\_stcd, 5) = 61650 THEN 'bier'

ELSE 'andere'

END FROM os20

WHERE os20\_artnr = '$1';

### Magnolia

#### Konfiguration

Die MySQL Abfrage schrieb ich in den neu erstellten ageCategory Node (Siehe Abbildung 22) vom Produktimporter Konfigurationsnode, der sich auf diesem Pfad befindet:

config:/modules/exchange/imports/produkte/nodeData/ageCategory

Ausgegraut wurden die bestehenden Knotenpunkte.

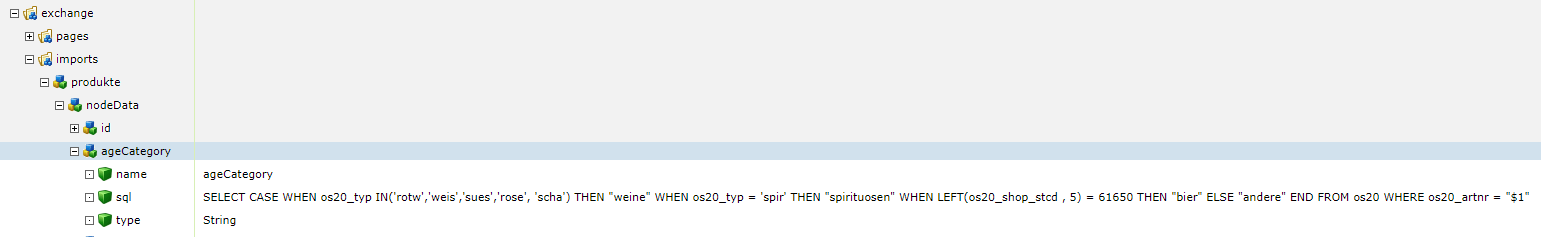


Abbildung ageCategory Produktimport

#### JCR

Diese Konfiguration bezweckt, dass beim Produktimport jedem importiertem Produktelement die Eigenschaft ageCategory mit der zugehörigen Alterskategorie zugewiesen wird (Siehe Abbildung 23). Der abgebildete Node liegt auf diesem Pfad:

shop:/products/barbera-dalba-scarrone-vigna-vecchia\_10021118

Ausgegraut wurden die bestehenden Knotenpunkte.

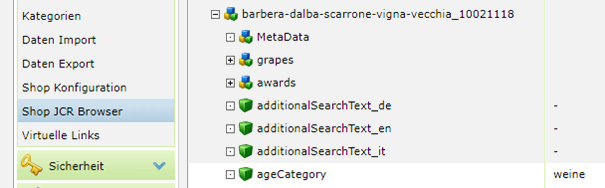


Abbildung ageCategory Produkt JCR

## Basket Rule

Um den Verkauf von alkoholhaltigen Getränken an Minderjährige effektiv zu verhindern, musste ein neues BasketRule und die dazugehörige Klasse BlockUnderage.java erstellt werden.

### Konfiguration

Im Konfigurationsknotenpunkt des block-underage BasketRule wird der Pfad zum Backend hinzugefügt sowie die Mitteilungen, die erscheinen sollen, falls das Alter noch nicht verifiziert ist oder man zu jung ist. Die Mitteilungen wurden nicht aus dem .properties File sondern direkt ins JCR geschrieben, da es dadurch in der Zukunft einfacher sein wird, die Werte abzuändern falls nötig. (Siehe Abbildung 24) Dieser Node befindet sich auf folgendem Pfad:

config:modules/shop/settings/minAge/block-underage/check-age

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Abbildung BasketRule Konfigurationsnode

### Java

#### BlockUnderage

Das BlockUnderage.java enthält die Logik, mit welcher überprüft werden soll, ob ein User genug alt ist, um die Produkte im Warenkorb zu kaufen. Hier wird zuerst die Variable minAge initialisiert. Falls diese nicht über 0 ist, dann braucht es keine weiteren Überprüfungen. Falls schon, werden die messages aus dem JCR in der Sprache ausgelesen, die gegenwärtig ausgewählt ist. Dann wird das gespeicherte Alter des Endbenutzers mit dem jetzigen Datum verglichen, um das Alter in Jahren zu erhalten, welches schlussendlich mit dem Mindestalter abgeglichen wird. Das Resultat kann eine der Beiden Fehlermeldungen oder die Weiterführung des Bestellverlaufes beinhalten.

Beim Abgleichen des Geburtsdatums des Users mit dem Jetzigen Datum, um das Alter zu erhalten stiess ich auf ein Problem. Nämlich wird bei der Konvertierung vom String verifiedDate im Format «yyMMdd» ins LocalDate userDate immer das jetzige Jahrhundert hinzugefügt (Beispiel: «880101» wird zu 2088-01-01 anstatt 1988-01-01. Ich dachte mir, dass die sicherste Methode, um das korrekte volle Jahr zu erhalten, eine Abfrage auf das jetzige Jahr sei. Falls das Geburtsdatum nach dem jetzigen Datum ist, dann werden dem Geburtsdatum 100 Jahre abgezogen.

LocalDate userDate = LocalDate.parse(verifiedDate, DateTimeFormatter.ofPattern("yyMMdd"));

LocalDate currentDate = LocalDate.now();

if(userDate.isAfter(currentDate)) {

    userDate = userDate.withYear(userDate.getYear()-100);

}

#### BasketFactory

Die Klasse BasketFactory.java war schon existent, ich habe aber die statische Methode getMinAge() neu erstellt, auf welche vom BlockUnderage.java und IDAgeVerificationModel.java zugegriffen wird. Hier wird zuerst durch die Produkte im Warenkorb iteriert, das höchste Mindestalter einer dem Produkt zugewiesener Kategorie in der Variable minAge gespeichert und schlussendlich retourniert.

## ShopUserPage

Damit der Kunde das Alter des Users manuell setzen kann, musste diese Option im Magnolia unter der Rubrik Shop > Users erstellt werden.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Abbildung EditUser Page

### HTML

Im ShopUserPage.html Datei musste ein Feld hinzugefügt werden, nämlich die Eingabe des verifizierten Geburtsdatums. Im Input Element habe ich mit einer Regular Expression sichergestellt, dass die Eingabe nur aus sechs Zahlen bestehen kann.

<tr>

    <td>Verifiziertes Geburtsdatum JJMMTT</td>

    <td><input type="text" name="verifiedDate" value="${userMap.user.verifiedDate!?html}" placeholder="980625" pattern="[0-9]{6}"></td>

</tr>

### Java

#### ShopUserPage

Um das Überprüfte alter eines Users manuell ins JCR durch die User-Seite zu speichern, benötigte es in der dazugehörigen Java-Klasse ShopUserPage.java diesen Codeblock.

if(StringUtils.isNotBlank(MgnlContext.getParameter("verifiedDate"))){

    userMap.put("verifiedDate", MgnlContext.getParameter("verifiedDate"));

}

#### UserFactory

Aus ShopUserPage.java wird die Methode UserFactory.updateUser() ausgeführt. Da diese Methode im UserFactory.java keine Property speichert, die noch nicht existiert, überlegte ich mir 2 Wege, dieses Problem zu lösen. Variante 1 wäre ein Import eines leeren Property verifiedDate in alle user Nodes, Variante 2 wäre die Einbindung des weiter unten eingefügten, mit Kommentaren gekennzeichneten, Codes.

Ich entschied mich für Variante 2, da es zwar nicht so «sauber» ist, jedoch mit alten Versionen der Shops kompatibel ist, bei welchen noch keine Eigenschaft verifiedDate im user Node existiert.

if(userConfig.hasProperty(userMapEntry.getKey())){

    manager.setProperty(user, userMapEntry.getKey(), (String)userMapEntry.getValue());

}else if(userMapEntry.getKey().equals("pswd")){

    manager.changePassword(user, (String)userMapEntry.getValue());

}

// Beginnn eingefügter code

else if(userMapEntry.getKey().equals("verifiedDate")){

    manager.setProperty(user, userMapEntry.getKey(), (String)userMapEntry.getValue());

}

// Ende eingefügter code

else{

log.info("wont save property \""+userMapEntry.getKey()+"\", no entry for it under /config/modules/shop/settings/userProperties");

}

## Bootstrap

Damit die Ressourcen, die nicht im Code bestehen (also JCR Nodes) beim Updaten eines Shops auch vorhanden sind, müssen diese als .xml Dateien ins Bootstrap und dem ShopModuleVersionHandler.java hinzugefügt werden. Dafür habe ich eine neue Version des Shop-Moduls erstellt.

### Ressourcen

In dem dazugehörigen Ordner alle neu erstellten Nodes, die ich beim Erstellen als .xml Dateien exportiert hatte, reinkopiert (Siehe Abbildung 26). Die zwei untersten Dateien (webseite.\*.xml) wurden in dieser Dokumentation noch nicht explizit erwähnt. Diese sind dafür da, die Komponente IDAgeVerification samt dem Kontaktformular beim Bootstrap-Prozess direkt auf den Seiten «Bestellübersicht» und «Konto Anpassen» einzufügen.

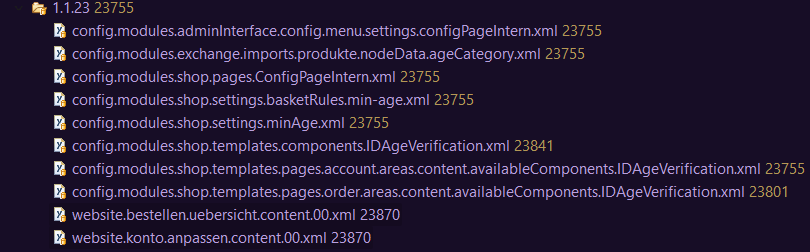


Abbildung Bootstrap

### VersionHandler

Im VersionHandler des Shop-Moduls erstellte ich ein neues Delta für die Version 1.1.23 samt den Tasks für jede oben erwähnte Ressource. Dazu verwendete ich die BootstrapReplaceOrCreate Klasse, welche früher von mir entwickelt wurde, um die Exception «Node with UUID xyz already exists», die bei der früher verwendeten Klasse BootstrapSingleResource auftauchte, zu verhindern.

# Kontrollieren

## Deploy Test

Nachdem ich mit dem Entwickeln fertig war, ging es weiter zur Kontrollphase. Dafür habe ich die webapp als ein WAR File exportiert und dieses mit einem shell script auf den Morpheus Server transferiert. Das Script hatte ich zu Beginn des Praktikums von meinem Arbeitskollegen erhalten, das einzige was ich abändern musste, war der name der webapp, der domain und der Ordners, wohin es transferiert werden soll. Das script beinhaltet auch Zeilen für das Stoppen und Wiederaufstarten der Webappp auf dem Server.

## Testprotokoll

Tabelle Beschreibung Testergebnisse

|  |  |
| --- | --- |
| OK | Test ist in Ordnung |
| HF | Hauptfehler: Behindert die Funktionsweise des Programms |
| NF | Nebenfehler: Behindert die Funktionsweise nicht |

Tabelle Testresultate

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Nr | Testfall | Browser | U1 | U2 |
| 1 | Bestellvorgang verifizierter User angemeldet |  | OK | OK |
| 2 | Bestellvorgang nicht verifizierter User angemeldet |  | OK | OK |
| 3 | Bestellvorgang verifizierter User anonymous |  | OK | OK |
| 4 | Bestellvorgang nicht verifizierter User anonymous |  | OK | OK |
| 5 | Anonymous Verifikationsvorgang ID korrekt |  | OK | OK |
| 6 | Anonymous Verifikationsvorgang Pass korrekt |  | OK | OK |
| 7 | Verifikationsvorgang eingeloggter User ID korrekt |  | OK | OK |
| 8 | Verifikationsvorgang eingeloggter User Pass korrekt |  | OK | OK |
| 9 | Verifikationsvorgang nicht korrekt |  | OK | OK |
| 10 | Bestellung ohne Altersverifikation |  | OK | OK |
| 11 | Bestellung mit Altersverifikation Ü16 |  | OK | OK |
| 12 | Bestellung mit Altersverifikation Ü18 |  | OK | OK |
| 13 | Alter setzen im Magnolia Shop > Users |  | OK | OK |
| 14 | Formularversendung manuelle Überprüfung |  | OK | OK |
| 15 | Anzeige Formular Bestellübersicht | B1 | OK | OK |
| B2 | OK | OK |
| B3 | OK | OK |
| B4 |  |  |
| 16 | Anzeige Formular Konto | B1 | OK | OK |
| B2 | OK | OK |
| B3 | OK | OK |
| B4 |  |  |

Die Testergebnisse zeigen, dass der Code so funktioniert wie er soll. Die Anzeige im Frontend erscheint auch plangemäss.

## Erläuterungen

Bei den Testfällen 1-14 wurde nur mit Browser B1 getestet, da der Prozess und somit das Resultat unabhängig vom Frontend ist. Mit B4 konnte ich leider nicht testen, da ich kein Apple Gerät besitze.

## Bootstrap-Test

Der Bootstrap-Test verlief ohne Fehler. Die zu Importierenden Ressourcen wurden im Magnolia JCR eingefügt und funktionieren nach Plan.

# Auswerten

## Reflexion zum Produkt

Insgesamt bin ich mit dem Produkt grösstenteils zufrieden. Ich konnte die geforderte Funktionalität des Auftrags innerhalb der geplanten Zeit erzielen. Es gibt jedoch ein paar Punkte, die ich gerne implementiert hätte, die Zeit dazu jedoch nicht gereicht hat.

Diese Erweiterungen wären möglich:

* Bestätigungsmeldung nach Absenden des Formulars, falls das Alter korrekt überprüft wurde (anstatt dem Verschwinden des Formulars)
* Direkte Anzeige des Kontaktformulars nach Fehlschlag der Verifikation durch Kennzahlen.
* Implementation der Altersverifikation im CustomView modul
* Automatisierter Import der Kategorien in das minAge JCR anstatt dem hardgecodeten Ansatz in der ConfigPageIntern.html.

## Reflexion zum Prozess

### Informieren

Diese Phase verging sehr gut, ich fing strukturiert und geordnet an, an der IPA zu arbeiten. Die Gründlichkeit, mit der ich die erste Phase durchgeführt hatte, half mir in den folgenden Prozessen.

### Planen

Das Planen war für mich ungewohnt und ich war deswegen auch eher unsicher unterwegs. Zwar verlief es auf den ersten Blick nicht schlecht, jedoch merkte ich beim Entwickeln, dass ich tiefgründiger hätte planen können. Dazu bereiteten mir die Diagramme und Testfälle Schwierigkeiten, da ich seit dessen Einführung während der Schulzeit mich nicht mehr damit befasst hatte.

### Entscheiden

Die Entscheidungen, die ich treffen musste, waren zum Glück leicht und ich bin mir sicher, dass es die richtigen waren.

### Realisieren

Das Realisieren hat mir am meisten gefallen und ging auch in meinen Augen neben dem Informieren am besten voran. Ich schätzte es, bei diesem Auftrag mehr mit dem Backend beschäftigt sein zu dürfen. Das Einzige, was mich ein Bisschen behindert hat, ist wie erwähnt die nicht genug detailliert abgeschlossene Planungsphase.

### Kontrollieren

Das Kontrollieren war für mich sehr mühsam, da es sehr repetitiv und wenig fördernd ist. Deswegen brauchte ich auch viel Kraft, mich darauf zu fokussieren und brauchte schlussendlich mehr Zeit, als dafür geplant war.

## Persönliches Fazit

Ich bin mit dem Endprodukt und dem Verlauf der IPA zufrieden. Der Auftrag hat mich motiviert, vor allem das Entwickeln des Backends. Ich konnte durch dieses Projekt meine Stärken klarer erkennen und gezielt an meinen Schwächen arbeiten. Es freut mich, ein Produkt zu liefern, welches in den Webshops bald integriert werden kann und somit der ständigen Entwicklung dieser beizutragen.

Inhaltsinformationen

# Abkürzungsverzeichnis

Tabelle Abkürzungsverzeichnis

|  |  |
| --- | --- |
| ADHS | Aufmerksamkeitsdefizit-/Hyperaktivitätsstörung |
| CMS | Content Management System |
| CSS | Cascading Style Sheets |
| EDV | Elektronische Datenverarbeitung |
| ERP | Enterprise Resource Planning |
| FTL | FreeMarker Template |
| HTML | HyperText Markup Language |
| IPA | Individuelle Praktische Arbeit |
| IPERKA | Informieren, Planen, Entscheiden, Realisieren, Kontrollieren, Auswerten |
| JCR | Java Content Repositories |
| NAS | Network Attached Storage |
| SASS | Syntatically Awesome Style Sheets |
| SQL | Structured Query Language |
| SVG | Scalable Vector Graphics |
| SVN | Apache Subversion |
| TTF | TrueType Font |
| XML | Extensible Markup Language |

# Glossar

Tabelle Glossar

|  |  |
| --- | --- |
| AdminCentral | Die CMS Admin Seite, auf welcher man die Webseite konfigurieren kann |
| Aktivierung einer Seite | Übertragung der in der admin-Instanz vorgenommenen Änderungen auf die www-Instanz |
| Aktivitätsdiagramm | Ein Flussdiagramm, das die von einem System ausgeführten Aktivitäten abbildet. |
| Apache Tomcat 7 | Apache Tomcat ist ein Open-Source-Webserver und Webcontainer, der die Spezifikation für Jakarta Servlets (früher: Java Servlets) und Jakarta Server Pages (JSP, früher: JavaServer Pages) implementiert und es damit erlaubt, in Java geschriebene Web-Anwendungen auf Servlet- beziehungsweise JSP-Basis auszuführen. |
| BasketRule | BasketRules sind «Regeln» die erfüllt sein müssen, damit ein User ein Produkt kaufen kann oder zusätzliche Kosten am Schluss der Bestellung berechnet werden müssen (zum Beispiel Lieferkosten oder Rabatte). |
| Bootstrap | Die Technik des Ladens bestimmter Ressourcen in das Programm, meist gebraucht bei neuen Versionen, die zu den bestehenden Ressourcen zusätzliche brauchen. |
| Branch | Ein Branch, zu deutsch Zweig, ist eine Verzweigung zu einer neuen Version, so dass unterschiedliche Versionen parallel im selben Projekt weiterentwickelt werden können. |
| Commit | Als Commit oder Check-In bezeichnet man bei der Verwendung von Versionsverwaltungssystemen den Vorgang des Einspie-lens von neuem oder geändertem Quelltext und ande-rer Dateien in das Versionsverwaltungssys-tem. |
| CustomView | Neues Modul der in der Inteco EDV AG entwickelten Webapps |
| Delta | Ein Delta repräsentiert die Unterschiede von einer Version eines Moduls zu einer anderen. Es enthält eine Reihe von Aufgaben und sollte eine Beschreibung dessen enthalten, was es tut. |
| Dialog | Konfigurationsfenster einer Komponente, welches im Bearbeitungsmodus erscheint. |
| Draw.io | Programm für das Erstellen von Diagrammen |
| Eclipse | Entwicklerumgebung |
| Freemarker | FreeMarker ist eine Template-Engine für die Java-Plattform, die Platzhalter in Textdateien durch dynamische Inhalte ersetzt. |
| Hardcoded | Nicht dynamisch geschriebener Code mit festgesetzen Werten. |
| I18n | Internationalisierung, Mehrsprachigkeit |
| Java | Objektorientierte Programmiersprache |
| JavaScript | Scriptsprache für Webbrowser |
| JQuery | JavaScript-Bibliothek |
| Magnolia | CMS auf Java-Basis |
| Mockup | Anschauungsmodell |
| Monospace | Eine Schriftart mit fester und einheitlicher Zeichenbreite |
| MySQL | Datenbankverwaltungssystem |
| Node | Knotenpunkt, beinhaltet verschiedene Properties oder Nodes |
| Property | Eigenschaft / Attribut |
| Regular Expression | Ein regulärer Ausdruck (englisch regular expression, Abkür-zung RegExp oder Regex) ist in der theoretischen Informatik eine Zeichenkette, die der Beschreibung von Mengen von Zeichenketten mit Hilfe bestimmter syntaktischer Regeln dient. |
| Rendering Engine | Programm zur Visualisierung von HTML |
| Repository | Verzeichnis, um Codeversionen zu verwalten |
| Script | Kommandozeilenprogramm |
| Shell | Mensch-Maschine-Schnittstelle |
| Standartshop | Mit dem standart Shop Modul in der Inteco EDV AG entwickelter Webapp |
| Template | Vorlage |
| Veeam | Die Veeam Software Group ist ein IT-Unternehmen, das Software für Datensicherungs-, Katastrophenwiederherstellungs- und Virtualisierungsmanagement für virtuelle Umgebungen von VMware vSphere und Microsoft Hyper-V entwickelt. |
| Visio | Programm für das Erstellen von Diagrammen |
| Visual Studio code | Quelltext-Editor |
| WEGAS | ERP-Software entwickelt von Inteco EDV AG |

# Quellenverzeichnis

Tabelle Quellenverzeichnis

|  |  |
| --- | --- |
| Info Berechnung, Abbildung 9 Prüfziffern | <https://www.kleemans.ch/identit%C3%A4tskarte-berechnung-pr%C3%BCfziffer> |
| Abbildung 5 Galaxus Screenshot | <https://www.galaxus.ch/de/customeraccount/machinereadablepersonidentificationnew/8424089> |
| GesG | <https://www.lexfind.ch/fe/de/tol/22315/versions/200204/de> |
| Abbildung 1 Iperka | <https://www.bin.ch/fileadmin/Dateien/Pausenplatz/0104_Meine%20Schreinerlehre/IPERKA.png> |
| Roboto Mono Font | <https://fonts.google.com/specimen/Roboto+Mono?preview.text=%3E%3E%3EInput%3E%3E&preview.text_type=custom&category=Monospace> |
| User Icon | <https://uxwing.com/male-icon/> |
| Glossar Aktivitätsdiagramm | <https://www.lucidchart.com/pages/de/uml-aktivitatsdiagramme> |
| Glossar Apache Tomcat7 | <https://de.wikipedia.org/wiki/Apache_Tomcat> |
| Glossar Branch | <https://de.wikipedia.org/wiki/Versionsverwaltung> |
| Glossar Commit | <https://de.wikipedia.org/wiki/Commit> |
| Glossar Delta | <https://nexus.magnolia-cms.com/content/sites/magnolia.public.sites/ref/5.4/apidocs/info/magnolia/module/delta/package-summary.html> |
| Glossar Freemarker | <https://de.wikipedia.org/wiki/FreeMarker> |
| Glossar Regular Expression | <https://de.wikipedia.org/wiki/Regul%C3%A4rer_Ausdruck> |
| Glossar Veeam | <https://de.wikipedia.org/wiki/Veeam> |

# Abbildungen

[Abbildung 1 IPERKA-Modell 4](#_Toc129969529)

[Abbildung 2 Ordnerstruktur 5](#_Toc129969530)

[Abbildung 3 Taskliste 6](#_Toc129969531)

[Abbildung 4 Zeitplan 7](#_Toc129969532)

[Abbildung 5 Altersüberprüfung galaxus 20](#_Toc129969533)

[Abbildung 6 UseCase Diagramm 21](#_Toc129969534)

[Abbildung 7 Flussdiagramm 23](#_Toc129969535)

[Abbildung 8 Klassendiagramm IDAgeVerificationModel 24](file:///C:\Users\GeorgiyShevoroshkin\Documents\IPA\probeIPA_altersüberprüfung\ipa.docx#_Toc129969536)

[Abbildung 9 Kleemans Kennzahlen 24](file:///C:\Users\GeorgiyShevoroshkin\Documents\IPA\probeIPA_altersüberprüfung\ipa.docx#_Toc129969537)

[Abbildung 10 Design Formular 34](file:///C:\Users\GeorgiyShevoroshkin\Documents\IPA\probeIPA_altersüberprüfung\ipa.docx#_Toc129969538)

[Abbildung 11 IDAgeVerification Konfigurationsnode 35](#_Toc129969539)

[Abbildung 12 IDAgeVerification Komponentennode 35](#_Toc129969540)

[Abbildung 13 Anfrageformular 36](file:///C:\Users\GeorgiyShevoroshkin\Documents\IPA\probeIPA_altersüberprüfung\ipa.docx#_Toc129969541)

[Abbildung 14 FormEmail Dialog 36](file:///C:\Users\GeorgiyShevoroshkin\Documents\IPA\probeIPA_altersüberprüfung\ipa.docx#_Toc129969542)

[Abbildung 15 FormEmail Dialog Email1 36](#_Toc129969543)

[Abbildung 16 FormEmail Dialog Email2 37](#_Toc129969544)

[Abbildung 17 Mindestalter Inputs 37](file:///C:\Users\GeorgiyShevoroshkin\Documents\IPA\probeIPA_altersüberprüfung\ipa.docx#_Toc129969545)

[Abbildung 18 ConfiPageIntern Rubriknode 38](#_Toc129969546)

[Abbildung 19 ConfigPageIntern Konfigurationsnode 38](#_Toc129969547)

[Abbildung 20 ConfigPageIntern Berechtigungen 39](#_Toc129969548)

[Abbildung 21 minAge JCR 39](file:///C:\Users\GeorgiyShevoroshkin\Documents\IPA\probeIPA_altersüberprüfung\ipa.docx#_Toc129969549)

[Abbildung 22 ageCategory Produktimport 40](#_Toc129969550)

[Abbildung 23 ageCategory Produkt JCR 40](#_Toc129969551)

[Abbildung 24 BasketRule Konfigurationsnode 41](#_Toc129969552)

[Abbildung 25 EditUser Page 42](#_Toc129969553)

[Abbildung 26 Bootstrap 43](#_Toc129969554)

# Tabellen

[Tabelle 1 Projektaufbauorganisation 3](#_Toc129945968)

[Tabelle 2 Vorkenntnisse 3](#_Toc129945969)

[Tabelle 3 Arbeitsprotokoll - Tag 1 8](#_Toc129945970)

[Tabelle 4 Arbeitsprotokoll - Tag 2 9](#_Toc129945971)

[Tabelle 5 Arbeitsprotokoll - Tag 3 10](#_Toc129945972)

[Tabelle 6 Arbeitsprotokoll - Tag 4 11](#_Toc129945973)

[Tabelle 7 Arbeitsprotokoll - Tag 5 12](#_Toc129945974)

[Tabelle 8 Arbeitsprotokoll - Tag 6 13](#_Toc129945975)

[Tabelle 9 Arbeitsprotokoll - Tag 7 14](#_Toc129945976)

[Tabelle 10 Arbeitsprotokoll - Tag 8 15](#_Toc129945977)

[Tabelle 11 Arbeitsprotokoll - Tag 9 16](#_Toc129945978)

[Tabelle 12 Arbeitsprotokoll - Tag 10 17](#_Toc129945979)

[Tabelle 13 Use Case 1 21](#_Toc129945980)

[Tabelle 14 Use Case 2 22](#_Toc129945981)

[Tabelle 15 Use Case 3 22](#_Toc129945982)

[Tabelle 16 Use Case 4 22](#_Toc129945983)

[Tabelle 17 Use Case 5 22](#_Toc129945984)

[Tabelle 18 Use Case 6 22](#_Toc129945985)

[Tabelle 19 Use Case 7 23](#_Toc129945986)

[Tabelle 20 Use Case 8 23](#_Toc129945987)

[Tabelle 21 Use Case 9 23](#_Toc129945988)

[Tabelle 22 Testbrowser 24](#_Toc129945989)

[Tabelle 23 Testumgebungen 24](#_Toc129945990)

[Tabelle 24 Testfall 1 25](#_Toc129945991)

[Tabelle 25 Testfall 2 25](#_Toc129945992)

[Tabelle 26 Testfall 3 26](#_Toc129945993)

[Tabelle 27 Testfall 4 26](#_Toc129945994)

[Tabelle 28 Testfall 5 26](#_Toc129945995)

[Tabelle 29 Testfall 6 27](#_Toc129945996)

[Tabelle 30 Testfall 7 27](#_Toc129945997)

[Tabelle 31 Testfall 8 28](#_Toc129945998)

[Tabelle 32 Testfall 9 28](#_Toc129945999)

[Tabelle 33 Testfall 10 29](#_Toc129946000)

[Tabelle 34 Testfall 11 29](#_Toc129946001)

[Tabelle 35 Testfall 12 29](#_Toc129946002)

[Tabelle 36 Testfall 13 30](#_Toc129946003)

[Tabelle 37 Testfall 14 30](#_Toc129946004)

[Tabelle 38 Testfall 15 30](#_Toc129946005)

[Tabelle 39 Testfall 16 31](#_Toc129946006)

[Tabelle 40 Beschreibung Testergebnisse 44](#_Toc129946007)

[Tabelle 41 Testresultate 44](#_Toc129946008)

[Tabelle 42 Abkürzungsverzeichnis 46](#_Toc129946009)

[Tabelle 43 Glossar 46](#_Toc129946010)

[Tabelle 44 Quellenverzeichnis 48](#_Toc129946011)

# Erläuterung zum Anhang

Dateien wurden neu erstellt:

BlockUnderage.java

ConfigPageIntern.java

IDAgeVerification.ftl

IDAgeVerification.css

IDAgeVerification.js

IDAgeVerificationModel.java

Schon existente Dateien, die bearbeitet wurden (beinhalten nur neu hinzugefügten Inhalt):

messages\_de.properties

messages\_en.properties

messages\_fr.properties

messages\_it.properties

BasketFactory.java

SessionSettingsFactory.java

ShopModuleVersionHandler.java

ShopUserPage.html

TemplatingFunctions.java

Schon existente Dateien, die bearbeitet wurden (Bearbeitungen sind kommentiert):

ShopUserPage.java

UserFactory.java

ConfigPageIntern.html

Code, der innerhalb vom Magnolia CMS gebraucht wurde:

ageCategoryImport.sql

formEmail1.ftl

formEmail2.ftl