

# Shopware 6 Erweiterung für Online-Shop

## Dokumentation

### **Berufsbezeichnung**

Fachinformatikerin Anwendungsentwicklung

### **Radosveta Galani**

Swebenbrunnen 15

22159 Hamburg

Prüflingsnummer: 131 54062

### **Ausbildungsbetrieb**

BFW Berufsförderungswerk Hamburg GmbH

Marie-Bautz-Weg 13

22159 Hamburg

### **Projektbetreuer**

Dr. Olaf Kubillus

E-Mail: [olaf.kubillus@bfw-hamburg.de](mailto:olaf.kubillus@bfw-hamburg.de)

Telefonnummer: +49 (0)40 64581-1233

# Inhaltsverzeichnis

1. Einführung.....	1
2. Projektdefinition.....	1
2.1. Ist-Analyse.....	1
2.2. Anforderungsdefinition (Soll-Konzept).....	2
3. Projektplanung.....	2
3.1. Ressourcenplanung.....	2
3.2. Kosten/Nutzen Analyse.....	3
4. Projektdurchführung.....	4
4.1. Shopare 6 als Shopsystem.....	4
4.2. Plugin Grundgerüst erstellen .....	4
4.3. Zusatzfelder und E-Mail-Vorlage erstellen.....	6
4.4. MHD im Storefront anzeigen.....	6
4.5. Produkte als TIPP anzeigen.....	9
4.6. Services.....	11
4.7. Subscriber.....	12
5. Tests .....	13
6. Soll- / Ist- Vergleich.....	14
7. Fazit.....	15
8. Glossar .....	15
9. Anhang.....	16
9.1. Information über Shopware 6.....	16
9.2. Datenbank.....	16
9.3. Teil der Tabelle – Produkt_Translation.....	18
9.4. Teil der Tabelle – Produkt .....	18
9.5. Plugin Code.....	19
9.6. Administration – Meine Erweiterungen.....	40
9.7. Administration – Mindesthaltbarkeit ZusatzfeldSet mit 2 Felder.....	40

<b>9.8. Suche nach Informationen.....</b>	<b>40</b>
<b>9.9. Profiler.....</b>	<b>41</b>
<b>9.10. ScheduledTask in der Datenbank.....</b>	<b>42</b>
<b>9.11. E-Mail, die der Admin bekommt.....</b>	<b>42</b>
<b>9.12. Anzeigen der Mindesthaltbarkeit nach der Lieferung für frische Produkte.....</b>	<b>43</b>
<b>9.13. Anzeigen des Mindesthaltbarkeitsdatums für langhaltbare Produkte.....</b>	<b>43</b>
<b>9.14. Anzeigen von TIPP Produkte.....</b>	<b>43</b>
<b>9.15. Anzeigen des Produktes im Shop bei ausgewählte Zusatzoption.....</b>	<b>44</b>
<b>9.16. Quellen.....</b>	<b>44</b>

# 1. Einführung

Die Firma „Vom Wege GmbH“ ist ein IT- Dienstleister mit Hauptsitz in Hamburg. Aktuell arbeiten dort 8 kreative Entwickler. Es ist ein wachsendes Unternehmen, das Online-Shops erstellt und betreut. Es werden innovative E-Commerce Lösungen, Web-Projekte jeder Art sowie eigene Produkte angeboten.

Die Konditorei Süßer Laden in Hamburg ist ein Kunde der Vom Wege GmbH. Sie bietet ihren Kunden täglich frisches Brot, Brötchen, ein großes Sortiment an Backwaren und süße Leckereien. Die Bäckerei hat die Kapazitäten große Mengen zu produzieren und außerhalb des Ladens zu verkaufen.

In den Lockdown Zeiten und im Moment wegen der Corona-Regelungen müssen 2 Meter Abstand zwischen den Kunden gehalten werden. Dies führt zu langen Schlangen auch außerhalb des Ladens, was beim schlechten Wetter unangenehm für die draußen Wartenden werden kann.

Die Inhaberin Maria Müller möchte ihren Kunden nicht nur die beste Ware, sondern auch die beste Bedienung anbieten. Sie hat sich entschieden zusätzlich zum Laden auch einen Online-Shop aufzumachen. Auf diese Weise können Kunden zukünftig alles ganz bequem von Zuhause aus bestellen und sich liefern lassen. So werden Kunden, die nicht in Kontakt mit vielen Menschen kommen wollen, über den Shop sicher bestellen. Kunden aus ganz Deutschland können auch die Leckereien genießen. Es werden neben den frisch zubereiteten Produkten auch verpackte, länger haltbare Süßwaren eigener und fremder Produktion angeboten.

Die Vom Wege GmbH hat den Auftrag bekommen einen Online Shop zu erstellen. Im Folgenden werden ausschließlich männliche Bezeichner für benutzte Rollen verwendet.

## 2. Projektdefinition

### 2.1. Ist-Analyse

Die Vom Wege GmbH setzt ihre Kundenprojekte mit der kostenfreien Shopware 6 - Community-Version um. Diese Version genügt für den erfolgreichen Start in den E-Commerce. Bei Anforderungen nach Zusatzfunktionen kann die Community-Version durch Plugins ergänzt werden. Die kostenpflichtigen und leistungsstärkeren Shopware-Versionen „Professional“ bzw. „Enterprise“ kommen nicht in Frage, da sie ähnliche Dienstleistungen wie unser Unternehmen anbieten.

Für das aktuelle Projekt der Erweiterung wurde ein Gespräch mit dem Kunden geführt und daraus folgende Ausgangssituation zusammengefasst.

Alle Lebensmittel sind nach einer bestimmten Zeit nicht mehr genießbar. Für das Lebensmittelgeschäft ist das Mindesthaltbarkeitsdatum eine sehr wichtige Produktinformation, die gesetzlich auf der Verpackung und zur Produktinformation anzugeben ist.

Für den Auftraggeber ist es aus organisatorischer und Planungssicht wichtig zu wissen, wann Platz im Lager frei wäre, wann alte Ware entsorgt und neue Ware eingekauft werden soll. Anstehende Zahlungen können geplant werden.

Die verwendete Shopware 6 Version, bietet kein Eingabefeld für das Mindesthaltbarkeitsdatum für die Produkte an. Im Produktbeschreibungs-Feld könnte die Mindesthaltbarkeit eingegeben werden, leider sind in diesem Fall keine weiteren Auswertungen möglich und im Shop werden auch die abgelaufenen Produkte angezeigt. Es gibt auch keine Möglichkeit, den Auftraggeber über ablaufende Produkte zu benachrichtigen. Die Lieferzeit für die verpackten Waren ist zwischen 1 bis 3 Tage. Damit mindestens ein Tag zum Genießen bleibt, dürfen nur langhaltbare Produkte, deren Mindesthaltbarkeit länger als 4 Tage ist, im Shop verkauft werden.

Um die gewünschte Funktionalitäten für den „Süßer Laden“ - Shop zu erfüllen, ist die Entwicklung einer Erweiterung notwendig, die alle diese Probleme lösen kann.

Diese wird im folgendem Projekt entwickelt.

## 2.2. Anforderungsdefinition (Soll-Konzept)

Die geplante Erweiterung soll die folgende Anforderungen erfüllen:

- Im Administration muss ein Zusatzfeldset – Mindesthaltbarkeit für den Artikelbereich erstellt werden.
- In diesem Zusatzfeldset soll es zwei Felder geben, eins vom Datum-Datentyp für die langhaltbaren Produkte und eins vom Text-Datentyp für die frischen Produkte.
- das Mindesthaltbarkeitsdatum muss im Shop(Frontend) angezeigt werden.
- Es soll beachtet werden, dass es frische und langhaltbare Produkte gibt und das MHD an unterschiedlichen Plätzen anzuzeigen ist.
- Das Mindesthaltbarkeitsdatum soll für langhaltbare Produkte im Produktdetailfeld ausgegeben werden.
- Für die frischen Produkte soll die Mindesthaltbarkeit nach der Lieferung im Produktbeschreibungsfeld stehen.
- Langhaltbare Produkte, die in 14 Tagen ablaufen, müssen im Shop hervorgehoben und als TIPP vorgeschlagen werden.
- Die abgelaufenen Produkte sollen aus dem Verkauf genommen werden (deaktiviert werden).
- Der Auftraggeber muss täglich per Email benachrichtigt werden, ob es Produkte gibt, die in 4 oder 14 Tagen ablaufen.

## 3. Projektplanung

### 3.1. Ressourcenplanung

Als Ressource sind ein Laptop macOS mit Betriebssystem Mojave und ein Server mit der installierten Software Shopware 6 geplant. Als Entwicklungsumgebung ist PHPStorm und Firefox zu empfehlen.

Für die Kalkulation wurde eine Stundenplanung gemacht. Eine Person soll dem Projekt für die folgenden Arbeitsschritte zur Verfügung stehen.

#### 1. Planungsphase

Summe	8
-------	---

#### 2. Kosten-Nutzen Analyse

Summe	2
-------	---

#### 3. Frontend - Entwicklung

Summe	14
<b>4.Backend - Entwicklung</b>	
Summe	25
<b>5.Tests</b>	
Summe	7
<b>6. Soll-/ Ist-Vergleich</b>	
Summe	2
<b>7. Dokumentation</b>	
Summe	12
<b>8.Gesamtsumme.....</b>	<b>70</b>

### 3.2. Kosten/Nutzen- Analyse

Für die gesamte Online-Shoprealisierung wurde ein Betrag von 20 000 Euro vereinbart. Die Freischaltung des Onlineshops übernimmt die Vom Wege GmbH auf einem von der Firma betreuten Server inklusive einer Qualitätskontrolle aller Inhalte und Funktionen. Für die weitere kostenpflichtige Shop-Betreuung ist ein Dienstleistungsvertrag plus monatliche Hosting mit dem Auftraggeber unterschrieben, der über 5 Jahre geht.

Nach Recherchieren im Internet und unter [store.shopware.com/erweiterungen/](https://store.shopware.com/erweiterungen/), wo Shopware eine Vielzahl von kostenlosen und kostenpflichtigen Erweiterungen anbietet, wurde festgestellt, dass es kein Plugin gibt, das die gewünschten Kriterien erfüllt.

Das Entwickeln des Plugins ist notwendig für den Online-Shop, da der Auftraggeber sonst seine Ware nicht verkaufen darf.

Zur Berechnung der Personalkosten wird ein monatlicher Durchschnittsbruttoverdienst für den internen Programmierer von 3300 € verwendet. Das entspricht einem Bruttostundenlohn von 20 €. Dazu werden addiert 20% Sozialversicherungskosten (vom Bruttogehalt) und weitere Kosten, wie 13.tes Gehalt, Weiterbildung. Der Betrag wird mit 18 € pro Stunde festgesetzt. Die Summe dieser Kosten entspricht 38 € pro Stunde und wird mit dem Programmieraufwand von 48 Stunden multipliziert. Alle weiteren Kosten sind pauschal in den 20 000 € berücksichtigt.

Für die Berechnung der Personalkosten für einen externen Programmierer werden 55 € pro Stunde festgelegt. Hier werden alle 70 Stunden berechnet. Die Kostenangabe wurde mit dem Projektleiter geklärt.

Variante 1 – Plugin intern programmieren.

Variante 2 – Plugin extern programmieren.

<b>Varianten</b>	<b>Variante 1</b>	<b>Variante 2</b>
Gesamtkosten	1824 €	3850 €

Shopware 6 wird täglich weiterentwickelt und Updates können wöchentlich erfolgen. In diesem Zusammenhang sind dann auch Wartungsarbeiten am Plugin zu erwarten. Diese und die Konfigurationskosten sind mit dem monatlichen Dienstleistungsvertrag abgedeckt. Variante 1 ist die günstigste und wird genommen.

## 4. Projektdurchführung

### 4.1. Shopware 6 als Shopsystem

Als Shopsystem bezeichnet man die einem Onlineshop zugrunde liegende Software. Shopsysteme beinhalten ein Warenwirtschaftssystem, (das jederzeit eine Übersicht über die Warenbestände ermöglicht), eine Listing-Funktion (die die zum Verkauf angebotenen Produkte ansprechend präsentiert) und Zahlungsschnittstellen (die verschiedenste Zahlungsarten und Paymentdienstleister verbinden).

Der Shop wurde mit dem Shopware 6 Shopsystem realisiert. Mehr Information über Shopware 6 befindet sich im Anhang.

Die Shopware-Plattform besteht aus drei Bausteinen:

#### **Shopware core**

Der Core ist das Zentrum der Plattform und umfasst alle eCommerce-spezifischen Workflows und Ressourcen. Er ist als modularer, nicht streng geschichteter Monolith aufgebaut. Die Module werden kategorisiert und auf verschiedene Verzeichnisse verteilt.

Der Core (Kern) ist so konzipiert, dass er viel Erweiterbarkeit zulässt ohne auf Wartbarkeit oder strukturelle Integrität zu verzichten.

#### **Shopware admin**

Die Administration ist eine Single Page Application, die neben der REST-API-basierten Kommunikation mit dem Core eine umfangreiche Benutzeroberfläche bietet. Es handelt sich um ein interaktionsorientiertes System nach dem Vorbild der Web-Komponenten - realisiert mit Vue.js. Die Administration zeigt Entitäten der Core-Komponente an und übernimmt die Verwaltung dieser.

#### **Shopware storefront**

Die Storefront ist eine Web-Benutzeroberfläche, welche die Kundensicht und die Steuerung der Shopware 6 Sales Channels ermöglicht. Die Storefront-Komponente, setzt auf dem Core auf.

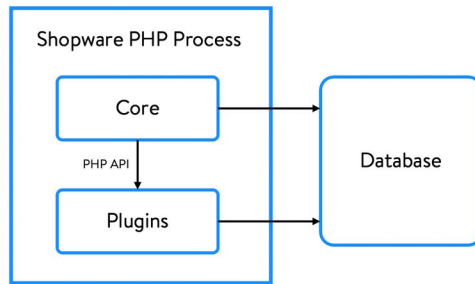
Da die Storefront als klassische PHP-Anwendung angesehen werden kann, verwendet sie HTML-Rendering, JavaScript und einen CSS-Präprozessor. Apropos Technologien: Die Storefront-Komponente verwendet Twig als Templating-Engine und SASS für Styling-Zwecke. Die Grundlage der Storefront-Vorlage basiert auf dem Bootstrap-Framework und ist daher vollständig anpassbar.

In diesem Projekt werden die Administrations- und Storefront-Bereiche erweitert werden.

### 4.2. Plugin Grundgerüst erstellen

Plugins sind der mächtigste Erweiterungsmechanismus, da sie verwendet werden können, um fast jeden Teil der Software zu erweitern, zu überschreiben oder zu modifizieren.

Plugins werden innerhalb des Shopware-Kernprozesses ausgeführt und können auf Ereignisse reagieren, benutzerdefinierten Code ausführen oder Dienste erweitern. Man hat direkten Zugriff auf die Datenbank.



Quelle: <https://www.shopware.com/de/>

Um ein Plugin zu programmieren ist es zwingend erforderlich sich an das Strukturschema von Shopware zu halten. Plugins werden innerhalb von Shopware in dem Ordner `custom\plugins` angelegt. Nur über diesen Pfad können Plugins verwendet werden.

Bei Abweichungen können Fehler bei der Ausführung von dem Plugin oder Shopware entstehen. Es wird in der DEV- Entwicklungsumgebung entwickelt, da alle Debugging Tools nur in dieser Entwicklungsumgebung aktiviert sind. Alle Environments teilen sich den gleichen Code, repräsentieren aber unterschiedliche Konfigurationen.

Mit dem CLI- Command

```
bin/console plugin:create SwagMindesthaltbarkeit
```

wird der Plugin - SwagMindesthaltbarkeit mit der folgenden Verzeichnisstruktur erstellt .



Die `composer.json` - Datei macht das Plugin funktionsfähig. Sie enthält grundlegende Informationen, die Shopware über das Plugin wissen muss, wie den technischen Namen, Beschreibung, Name, Lizenz, aktuelle Plugin- Version, erforderliche Abhängigkeiten und noch mehr. (siehe Anhang 9.5.17).

Ordner `src`-Dies ist nicht erforderlich, aber empfehlenswert. Hier befindet sich der ganze Code.

Die Benennung der Plugin-Basisklasse muss die gleiche sein wie der Plugin-Name.

Die Basisklasse `SwagMindesthaltbarkeit.php` kann die folgenden Methoden implementieren, um jede Art von Installations- oder Wartungsaufgaben auszuführen- `install()`, `update()`, `uninstall()`, `activate()`, `deactivate()`.

Service.xml

Alle Dienste, Commands, Subscribers müssen in der Datei `services.xml` registriert



werden, um von überall gesehen und genutzt werden zu können.. Diese Datei liegt in einem Verzeichnis namens `src/Resources/config/`.

### 4.3. Zusatzfelder und E-Mail-Vorlage erstellen

Für das Erstellen der neuen Zusatzfelder und der E-Mail-Vorlage werden zwei neue Datensätze in die Datenbank geschrieben - ein Zusatzfeldset und eine E-Mail – Vorlage. (Anhang). Damit das entwickelte Plugin mobil sein kann, werden alle benötigten Objekte für den Administrationsbereich in `SwagMindesthaltbarkeit.php` in der Funktion `install()` erstellt und mit der Funktion `uninstall()` werden dieselben Objekte beim Plugindeinstallieren gelöscht. (siehe Anhang 9.5.16).

Im Gegensatz zu den meisten Symfony-Anwendungen verwendet Shopware kein ORM (siehe Glossar), sondern eine dünne Abstraktionsschicht namens Data Abstraction Layer (DAL). Die DAL ist auf die spezifischen Bedürfnisse von Shopware abgestimmt und ermöglicht Entwicklern den Zugriff auf die Datenbank über vordefinierte Schnittstellen. Der zentrale Zugriffspunkt ist das Repository. (im Anhang und Glossar ist mehr über DAL zu erfahren).

In der Administration- Benutzeroberfläche werden beide Entities an den richtigen Stellen angezeigt – das ZusatzfeldSet (mit den beiden Zusatzfeldern) bei der Liste von ZusatzfeldSets und die E-Mail Vorlage bei der Liste von E-Mails Vorlagen. (Siehe Anhang 9.7.).

### 4.4. MHD im Storefront anzeigen

Als nächstes wird das Mindesthaltbarkeitsdatum (MHD) im Storefront angezeigt. Für die langhaltbaren Produkte soll das Datum am besten neben den Preis- und Lieferinformationen dargestellt werden, so dass es sofort vom Shop-Kunde gesehen wird. Für die frischen Produkte ist das Produktbeschreibungs-Feld die bessere Position, da diese nur eine weitere Erläuterung zum Produkt ist. Ein anderer Grund, der für das Produktbeschreibungs-Feld spricht, ist, dass eine lange Textbeschreibung das Produkt-Details-Feld unübersichtlicher machen würde.

Nachdem der Platz für das MHD festgelegt ist, soll der Code erweitert werden. Als Erstes wird der Shop-Benutzeroberfläche (Storefront) HTML- Code untersucht und der gesuchte Platz im Code ermittelt .



Dann werden die benötigten Informationen entnommen und mit Hilfe des Symfony Profilers (oder Suche im Code) im Shopware-Code gefunden.

Da Shopware auf dem Symfony-Framework aufbaut, gewährt es auch Zugriff auf den Symfony Profiler (siehe Glossar).

Mit dem Profiler werden die Twig-Dateien durchgesucht.

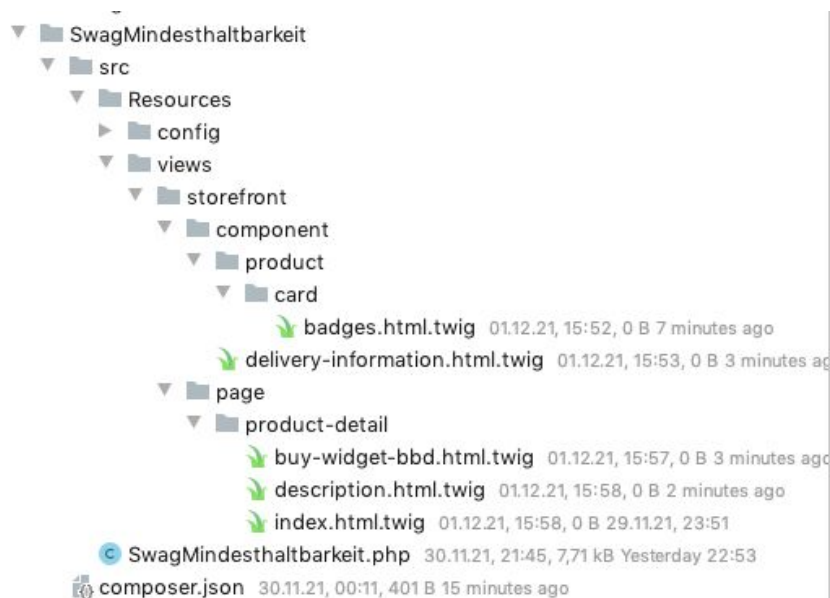
Danach werden die Dateien im Plugin an der richtigen Stelle neu erstellt und erweitert.

vendor/shopware/storefront/Resources/views/storefront/component/product/card/wishlist.html.twig		
📄 @Storefront/storefront/page/product-detail/tabs.html.twig		1
vendor/shopware/storefront/Resources/views/storefront/page/product-detail/tabs.html.twig		
📄 @SwagMindesthaltbarkeit/storefront/page/product-detail/description.html.twig		1
custom/plugins/SwagMindesthaltbarkeit/src/Resources/views/storefront/page/product-detail/description.html.twig		
📄 @Storefront/storefront/page/product-detail/description.html.twig		1
vendor/shopware/storefront/Resources/views/storefront/page/product-detail/description.html.twig		
📄 @Storefront/storefront/page/product-detail/review/review.html.twig		1
vendor/shopware/storefront/Resources/views/storefront/page/product-detail/review/review.html.twig		

Bild.1.

Um das MHD für frische Produkte anzeigen zu können, muss die Datei description.html.twig in dem Verzeichnis Resources/views/storefront/page/product-detail erweitert werden. Die blau markierte Datei description (siehe Bild.1.) ist die neu erstellte Datei im SwagMindesthaltbarkeit und die darunter liegende ist die Originaldatei, die erweitert wird.

Die Plugin- Struktur sieht zur Zeit so aus:



Für die langhaltbare Produkte soll die delivery-information.html.twig Datei erweitert werden, die im component Verzeichnis liegt.

Die Namen der erweiterten Dateien fangen immer mit **sw\_extends** an, gefolgt vom Namen der erweiterten Datei. In diesem Fall:

```
{% sw_extends '@Storefront/storefront/component
               /delivery-information.html.twig' %}
```

Danach folgen Twig Blöcke, die zusammen mit HTML die Seite aufbauen. Der neu entwickelte Block muss zwischen zwei Blöcken - Preisinformation und Lieferungsinformation - positioniert werden. Der geschriebene Block soll syntaktisch und strukturell an den original Shopware - Code angepasst werden. Kleine Blöcke können direkt zugefügt werden. Sonst wird der Code in einer separaten Datei gespeichert und mit sw\_include eingebunden.

In folgenden Zeilen stehen die Bedingungen unter denen das MHD angezeigt werden darf:

- wenn das Feld nicht leer ist
- wenn das Erscheinungsdatum nicht größer als heute ist
- wenn Lieferung im Moment möglich ist
- wenn das MHD-nach der Lieferung-Feld nicht ausgefüllt ist

```
{% if
page.product.translated.customFields.custom_best_before_date|
date('U') > ("now"|date('U') + 345600)

    and (product.releaseDate|date('U') < "now"|date('U' + 345600))
    and not general.deliveryNotAvailable
    and not
page.product.translated.customFields.custom_best_before_after_delivery %}
```

```
<p class="delivery-information date-info">

    <span class="delivery-status-indicator bg-info"></span>
    <strong>{{ "customFields.custom_best_before_date"|trans }}</strong>

    {{ page.product.translated.customFields.custom_best_before_date|
        format_date('long', locale=app.request.locale) }}
</p>
```

Der Paragraf <p> zeigt das MHD in folgendem Format an, wenn alle Bedingungen erfüllt sind: „ 5. Dezember 2021“

Alternativ, wenn das MHD <= heute + 4 ist, bleibt dieses Feld leer. Folgende Zeilen sind nicht notwendig aber wurden beim Tests genutzt.

```
{% elseif
page.product.translated.customFields.custom_best_before_date|
date('U') <= ("now"|date('U') + 345600) %}

    <p class="delivery-information date-info"></p>

{% endif %}
```

Die {{ parent() }}- Funktion gibt alle anderen Datei-Elemente an der Stelle aus.

Wenn ein Zusatzfeld erstellt wird, wird ein technischer Name gegeben. Für diesen wird ein Text in den Textbausteine im Bereich Administration gespeichert. Nach Eingabe des technischen Namens im Twig, wird im Frontend der Text aus dem Textbaustein ausgegeben. In dem Fall sieht er so aus:

```
Twig : {{ "customFields.custom_best_before_date"|trans }}
```

```
Shop : „ Mindesthaltbarkeitsdatum :“
```

trans ist ein Twig - Filter und wird für Mehrsprachigkeit benutzt. Dafür sollen Snippets für die Übersetzung erstellt werden.



Das sind json-Dateien, die zeigen, wie ein Entity mit bestimmten technischen Namen in unterschiedliche Sprachen übersetzt werden soll.

### swag\_mindesthaltbarkeit.de\_DE.json

```
{
  "customFields": {
    "custom_best_before_date": "Mindesthaltbarkeitsdatum: ",
    "custom_best_before_after_delivery": "Mindesthaltbarkeit nach der  
Lieferung : "
  }
}
```

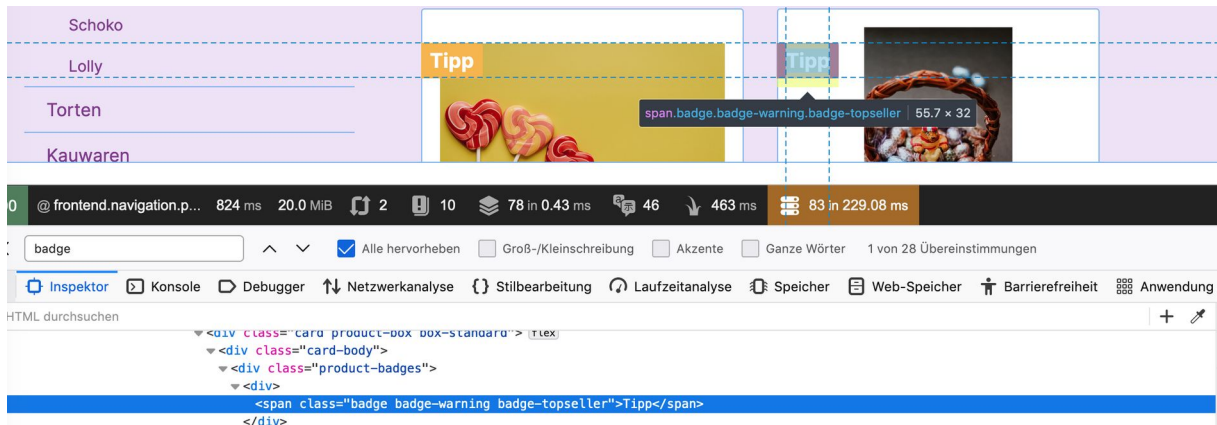
### swag\_mindesthaltbarkeit.en\_GB.json

```
{
  "customFields": {
    "custom_best_before_date": "Best before : ",
    "custom_best_before_after_delivery": "Best before after delivery : "
  }
}
```

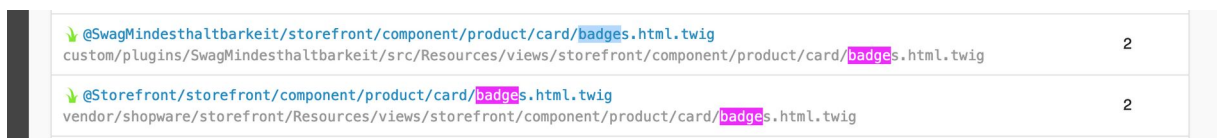
Der gesamte Code mit Bildern befindet sich im Anhang.

## 4.5. Produkte als TIPP anzeigen

Mit demselben Verfahren wie der MHD - Platz im HTML Code gefunden wurde, wird TIPP in Product – Card bei der Produktaufistung gesucht.



Danach wird die zu erweiternde Datei neu im Plugin erstellt und erweitert und an die Shopware Struktur und Semantik angepasst.



Die Datei badges.html.twig befindet sich im Verzeichnis storefront/component/product/card.

Die Bedingungen unter denen ein Produkt als TIPP markiert wird, sollen geändert und an die Projekt- Anforderungen angepasst werden. (Vollständiger Code im Anhang 9.5.6.)

```
{% sw_extends
 '@Storefront/storefront/component/product/card/badges.html.twig' %}
```

Unten ist der Block, der den TIPP - Artikel im Shop anzeigt und die folgende Anforderung erfüllen soll:

- wenn das MHD nach 14 Tagen abläuft soll das Produkt hervorgehoben werden (als TIPP angezeigt werden).

Die Tage werden in Sekunden angegeben (Formatstandard).

Anderenfalls wird mit der Funktion parent() der Originalcode ausgeführt.

```
{% block component_product_badges_topseller %}
    {% if product.translated.customFields.custom_best_before_date|
date('U') <= ("now"|date('U') + 1209600 )
        and product.translated.customFields.custom_best_before_date|
date('U') > ("now"|date('U') + 345600 ) %}

        <div>
            <span class="badge badge-warning badge-topseller"> {{
"listing.boxLabelTopseller"|trans|sw_sanitize }} </span>
        </div>
```

```
{% else %}

    {{ parent() }}

{% endif %}
{% endblock %}
```

**listing.boxLabelTopSeller** ist der gespeicherte Textbaustein, der im Shop als TIPP beim Anzeige des Produktes erscheint.

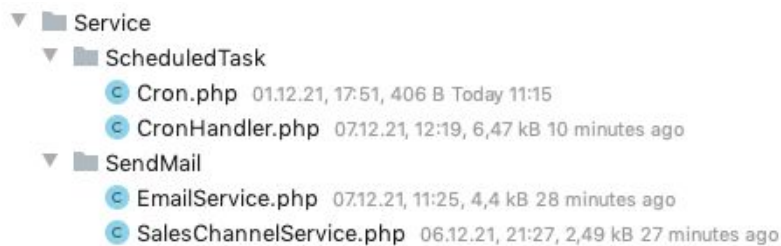
### 4.6. Services

Services sind im Idealfall nur eine einfache PHP-Klasse. Service-Dateien liegen in dem gleichnamigen Ordner Services. Die erstellte php-Datei sollte auf Service enden, dies erleichtert das Verständnis für andere Programmierer. Alle Services werden in der Datei service.xml aufgelistet und nach dem Plugin-Aktivieren registriert. Auf dieser Weise werden sie zur Verfügung gestellt. So kann überall in Shopware auf dieselbe Instanz dieser Klasse zugegriffen werden. In vielen Fällen können Dienste von anderen Diensten abhängen. Normalerweise werden diese als Eigenschaften über den Konstruktor hinzugefügt.

Es wird ein EmailSenden Service benötigt um Emails an den Admin zu senden. Ohne SalesChannelContext kann keine Email gesendet werden. Dafür wird dies in einem separaten Service erstellt. Die Trennung in kleinere Services wird gemacht, um diese bei Bedarf überall zu nutzen.

Um das tägliche Email-Versenden zu ermöglichen ist der Shopware Service ScheduledTask (Cronjob) vorgesehen. Der Cronjob automatisiert wiederholte Prozesse (Code der regelmäßig ausgeführt wird).

Beide Services ScheduledTask (Cronjob) und SendMail (EmailSenden) werden in demselben Verzeichnis Service abgelegt. Die Dateien SalesChannelService.php und EmailService.php liegen im SendMail Verzeichnis.(siehe Anhang).



Cron.php und CronHandler.php liegen in ScheduledTask Verzeichnis.

#### Cron.php

```
<?php
```

```
declare(strict_types=1);
namespace SwagMindesthaltbarkeit\Service\ScheduledTask;
use Shopware\Core\Framework\MessageQueue\ScheduledTask\ScheduledTask;
class Cron extends ScheduledTask
{
    public static function getTaskName(): string
```

```
{
    return 'swag.cron_task';
}
public static function getDefaultInterval(): int
{
    return 86400; // 24 Stunden in Sekunden
}
}
```

Diese Datei erweitert die Shopware - ScheduledTask Klasse. In der wird die Name und der Abstand (in Sekunden) der durchzuführenden Aufgabe definiert.

In CronHandler.php liegt der auszuführende Code für das Senden von Emails an den Admin. Das ist die Funktion `emailAnAdminSenden()`. (siehe Anhang). Die Funktion `run()` führt alle Funktionen aus, die ausgeführt werden müssen.

```
public function run(): void
{
    $this->emailAnAdminSenden();
}
```

Services von Shopware können durch extends erweitert werden oder wenn der Shopware Service als Eigenschaft in unserer Service - Klasse definiert ist.

Es wurde eine Klasse `SendEmailService.php` erstellt.

Als Eigenschaft wurde die `$mailService` Variable vom Typ `AbstractMailService` definiert.

Es wurde eine Instanz der Shopware Klasse `DataBag` initialisiert. Alle in `DataBag` gesetzten Daten sind zum Email-Erstellen notwendig und müssen in der Klasse, die diesen Service nutzt, wieder verwendet werden.

Es wird auch der `SalesChannelService` erstellt und in der Klasse `EmailService.php` implementiert. Es ist für das Erstelle von `SalesChannelContext` erforderlich, der als Parameter in den Funktion `send()` zu übergeben ist. (siehe Anhang 9.5.13).

### 4.7. Subscriber

Die Subscriber werden innerhalb des Plugins in dem Ordner `Subscriber` abgelegt. Subscriber sind Services welche sich auf ein Event subscriben und anschließend eine Funktion aufrufen. Um eigene Subscriber zu verwenden, müssen sie an die `services.xml` übergeben werden. Beim Aktivieren des Plugins werden die eigenen Subscriber im DI-Container registriert und werden bei dem jeweiligen Event verwendet.

Eine zusätzliche Funktion wurde entwickelt, die das Handeln mit den nach 4 Tagen ablaufenden Produkte betrifft. Es wurde gewünscht, dass die Produkte nicht mehr zum Verkauf im Shop angeboten werden (deaktiviert werden). Diese Funktion ist erfüllt, aber nach Wunsch kann man eine Funktion aktivieren, die das Produkt im Shop anzeigt, aber kein Kauf zulässt.

Dafür wurde noch ein zusätzliches Umschalt- Zusatzfeld erstellt und integriert. Als Default bleibt, dass das Produkt ausgeschaltet wird. Bei Aktivieren des Feldes wird das Produkt weiter im Shop zu sehen sein, nur die Option zum Kaufen wird es nicht geben.

Diese Funktionalität ist mit Subscriber ermöglicht. Der Subscriber wartet auf das `onPageLoadetEvent`. Bei diesem Event wird die Homepage geladen. Dann wird geprüft, welche Produkte in 4 Tagen ablaufen. Wenn die Zusatzoption gewählt wurde, dann wird das Produkt nicht deaktiviert und bleibt im Shop zu sehen.



Um die Kaufoption zu verbergen, muss die index.html.twig Datei erweitert werden.

In einem Block wird geprüft, ob das MHD  $\leq$  heute + 4 Tage ist und ob die zusätzliche Funktion - custom\_best\_before\_check == true ist. Wenn das der Fall ist wird die neu entwickelte Datei

```
@Storefront/storefront/page/product-detail/buy-widget-bbd.html.twig
```

hinzugefügt. Wenn nicht, bleibt die original Datei und das Produkt wird mit allen Produkt-Details und Kaufoption angezeigt.

## 5.Tests

Folgend wird die Plugin-Funktionalität mittels Black-Box-Testverfahren und White-Box-Testverfahren geprüft.

Das Plugin wurde in der DEVELOPMENT Umgebung entwickelt.

Der Profiler ist ein leistungsstarkes Entwicklungstool (nur in der DEV- Umgebung), das detaillierte Informationen über die Ausführung einer Anfrage liefert .

Dadurch ist es möglich schnell herauszufinden, warum die eigenen Dateien nicht geladen werden können. Die syntaktischen Fehler wurden sofort erkannt und sofort beseitigt.

Nachdem das MHD im Frontend angepasst und ausgegeben wurde, wurde mehrmals getestet, ob die Anforderungen erfüllt sind.

Es wurden Commands entworfen, die in CLI ausgeführt wurden und auf diese Weise die Funktionalität für alle Methoden geprüft. Es gibt Commands für die zwei Produkt-Listen sowie Command für das Email-Versenden (Bilder im Anhang).

### Test 1

Es wurden mehrere Produkte erstellt und für alle unterschiedliche MHD angegeben:

1. Produkt hatte MHD = 4 Tage
2. Produkt hatte MHD < 4 Tage
3. Produkt hatte MHD > 4 Tage
4. Produkt hatte MHD  $\leq$  14 Tage TIPP anzeigen
5. Produkt hatte MHD in beiden Feldern, für langhaltbare und für frische Produkte gleichzeitig ausgefüllt
6. Produkt hatte MHD < als Erscheinungsdatum

Es wurden die erwarteten Ergebnisse bekommen. Das MHD wurde nur dann angezeigt, wenn MHD > als 4 Tage war und wurde als TIPP markiert, wenn MHD  $\leq$  14 Tage.

Wenn beide Felder ausgefüllt wurden, wurde nur die Textbeschreibung angezeigt (wie erwartet).

### Test 2

Um die Listen mit Produkten, die nach 4 und 14 Tagen ablaufen, zu prüfen, wurden Commands zum sofort testen erstellt.

Beim ersten Test wurde eine lange Liste angezeigt, die nicht nur die Produkte enthielt, die nach 4 Tage ablaufen sondern alle Produkte, die bis zum angegebenen Datum ablaufen. Es wurden auch Produkte angezeigt, die vor 2 Monaten abgelaufen sind.

Dieses Problem wurde mit einer Code Änderung beseitigt.

### Test 3



Nachdem der Cronjob erstellt und registriert wurde, wurde die Funktionalität getestet. Obwohl das Command für Emailversenden funktioniert hatte, hat die regelmäßige Benachrichtigung nicht funktioniert. Ein kurzer Blick in die Datenbank zeigte, dass als nächste Ausführungszeit eine Millisekunde vor der Registrierungszeit gespeichert ist. Nach einer manuellen Änderung hat es trotzdem nicht funktioniert.

Shopware\Core\Checkout\Cart\Cleanup\CleanupCartTask	86400	scheduled	2021-12-07 12:09:26.626	2021-12-07 12:09:26.626	2021-05-28 11:03:16.648
SwagMindesthaltbarkeit\Service\ScheduledTask\Cron	86400	scheduled	NULL	2021-12-07 12:12:00.836	2021-12-07 12:12:00.837
Shopware\Core\Framework\Adapter\Cache\InvalidateCacheTask	20	queued	2021-08-09 09:29:16.701	2021-08-09 11:05:21.222	2021-05-28 11:03:16.623

Nach Recherchieren in Shopware - Foren wurde schnell klar, dass dieses Problem nach einem Update vor 6 Monaten öfter aufgetreten ist und zur Zeit daran gearbeitet wird. Dann wurde eine kurzfristige Lösung gefunden, indem die Cronjob - Funktionalität des Servers genutzt wurde. Das Command `swag-commands:email` wurde zum regelmäßigen Ausführen am Server registriert. Es funktioniert alles korrekt.

```
# m h dom mon dow   command
40 13 * * * /var/www/shopware/bin/console swag-commands:email
```

### Test 4

Mit derselben Funktion, die die Liste mit den Produkten erstellt, die in 4 Tagen ablaufen werden auch diese Produkte gleichzeitig deaktiviert. Diese Funktion wurde mehrmals getestet, indem die Produkte nacheinander aus und angeschaltet wurden. Alles funktionierte einwandfrei.

### Test 5

Die Zusatz Funktion wurde auch getestet. Für 2 Produkte wurde die Zusatzfunktion gewählt und für 2 andere nicht. Dann, nach dem Ausschalten der Produkte, wurden diese weiter im Shop angezeigt mit der Info, dass die zur Zeit nicht zum Verkauf stehen.

## 6. Soll- / Ist-Vergleich

Die Erweiterung soll erlauben, Waren mit einem Mindesthaltbarkeitsdatum (MHD) zu verwalten.

Bei der Produkterstellung sollte es im Administrationsbereich neue Zusatzfelder geben, in denen das Mindesthaltbarkeitsdatum für langhaltbare und frische Produkte eingetragen werden kann.

Diese Felder sind erstellt und erlauben für langhaltbare Produkte, ein Datum aus dem Kalender auszuwählen und für frische Produkte in einem Textfeld, die Aufbewahrungsbedingungen und die Tage zu beschreiben.

Das Mindesthaltbarkeitsdatum sollte im Shop bei den Produkt Lieferinformationen angezeigt werden. Dieses Kriterium ist erfüllt. Für die frischen Produkte ist dieses Kriterium auch erfüllt bei der Produktinformation.

Es sollte der Auftraggeber(Admin) täglich per Email benachrichtigt werden, welche Produkte in 4 Tage und in 14 Tage ablaufen. Diese Anforderung ist auch erfüllt. Der Admin bekommt täglich eine Email mit zwei Listen, eine mit den in 4 Tagen ablaufenden und die zweite mit den in 14 Tagen ablaufenden Produkten. Dafür wurde eine spezielle Email-Vorlage erstellt.

Es sollten die Produkte, die in 14 Tagen ablaufen, als TIPP angezeigt werden. Diese Anforderung ist auch erfüllt. Jedes Mal wird bei der Produkte – Auflistung geprüft, ob das Produkt TIPP erhalten soll.

Die nächste Anforderung, die Produkte, die in 4 Tagen ablaufen, dürfen nicht verkauft werden, ist auch erfüllt. Gleichzeitig mit dem Email-Versenden werden die Produkte aus der Liste mit den Produkten, die in 4 Tagen ablaufen, sofort deaktiviert und sind nicht mehr im Online-Shop zu sehen.

Alle Anforderungen wurden erfüllt. Bilder der Ergebnisse stehen im Anhang.

Es wurde eine zusätzliche Funktion angeboten. Diese erlaubt nach Wunsch dass die Produktinformationen im Online-Shop angezeigt werden, aber das Produkt selbst nicht zum Verkauf angeboten wird.

Die kalkulierte Zeit war passend. Fast alle Phasen haben sich überlappt. Bei der Frontend-Entwicklung wurde auch gleichzeitig getestet. Parallel mit der Backend-Entwicklung wurden die Funktionen getestet, ob sie das gewünschte Ergebnis zurückgeben. Die Planungsphase war kürzer. Dadurch gab es mehr Zeit um noch eine zusätzliche Funktionalität zu entwickeln. Aus gesundheitlichen Gründen wurde die Dokumentationsphase mehrmals unterbrochen und durfte vielleicht etwas länger sein. Die Soll/Ist Analyse war schnell und der Rest der Zeit wurde für das Schreiben der Dokumentation genutzt.

## 7. Fazit

Shopware 6 ist ein relativ neues Produkt der Shopware AG. Es hat eine große Community, die sehr schnell viele neue Features entwickelt. So bekommt Shopware 6 sehr oft Updates, die nicht immer mit allen Plugins kompatibel sind. Das verlangt neue Quellcode-Anpassungen und kann manchmal viel Zeit kosten.

Trotz allem ist Shopware 6 ein sehr angenehmes Shopsystem, mit dem ich sehr gerne arbeite. Es gibt immer etwas Neues zu lernen. Der Administrations-Bereich ist sehr angenehm und strukturiert.

Das Plugin ist gut und sehr hilfreich geworden. Es sind keine offenen Punkte geblieben. Man kann immer etwas nach Kundenwunsch mit der Zeit nachbessern.

## 8. Glossar

**Symfony Profiler** - Mit dem Profiler können alle ausgelösten Ereignisse in der aktuellen Anfrage leicht gefunden werden, einschließlich ihres jeweiligen zu verwendenden Namens. . Wann immer es ein Problem mit dem Code gibt, sieht man eine Fehlerseite, die alles zeigt, was man braucht, um das Problem zu verstehen und herauszufinden woher es kommt.

**ORM-** Objektrelationale Abbildung (englisch object-relational mapping, ORM) ist eine Technik der Softwareentwicklung, mit der ein in einer objektorientierten Programmiersprache geschriebenes Anwendungsprogramm seine Objekte in einer relationalen Datenbank ablegen kann. Dem Programm erscheint die Datenbank dann als objektorientierte Datenbank, was die Programmierung erleichtert.

**DAL-** Mit Shopware 6 kam der neue Data Abstraction Layer (DAL) hinzu, Shopwares eigene Abstraktionsschicht. Durch die neue Entwicklung hat man Zugriff auf alle notwendigen Daten einer Datenbank.

Wenn etwas über die DAL-Suche gesucht wird, bekommt man eine Sammlung an Daten zurück, die nach bestimmten Kriterien sortiert werden können. Des Weiteren werden Metainformationen basierend auf der Suche zurückgegeben.

Wie bei Doctrine ORM ist der zentrale Zugriffspunkt das Repository. Dies benutzt man, um mit der hinterlegten Storage-Engine zu kommunizieren. Das Repository beinhaltet alle Basismethoden. Dadurch kann mit den gespeicherten Daten arbeiten. Um das Verarbeiten innerhalb Shopwares einfacher zu machen, löst jeder DAL-Request ein eigenes Event aus. Egal was gemacht wird, ob Daten auslesen oder Daten hinzufügen, alles löst ein eigens Event aus, welches Shopware verarbeiten kann. Es gibt jeweils eine Implementierung für einen Writer, Reader, Searcher und Aggregation.

**DI-Container** - Dependency Injection Container

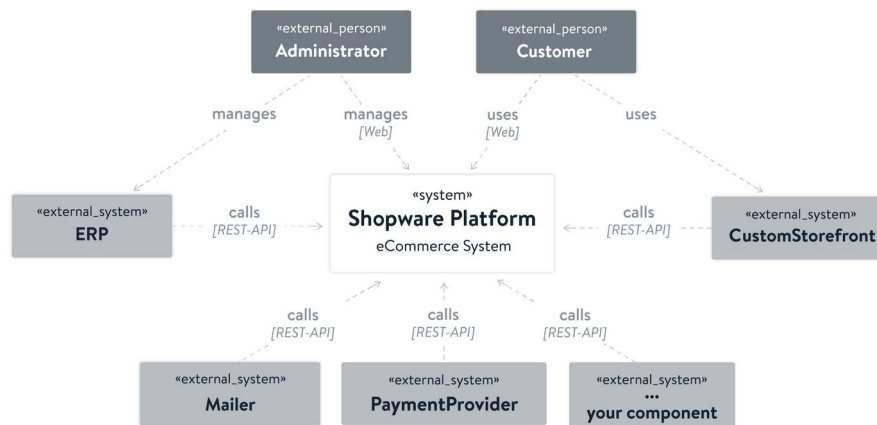
**MHD** - Mindesthaltbarkeitsdatum

## 9. Anhang

### 9.1. Information über Shopware 6

Shopware 6 ist eine Open-Source eCommerce-Plattform, die aus den Ideen und dem Geist ihrer Community besteht. Shopware 6 ist sehr komplex und leistungsfähig. Immer mehr Shopbetreiber – insbesondere in Deutschland - setzen auf das in Schöppingen entwickelte System Shopware. das diese Technologien verwendet:

- Symfony 4.2
- Twig
- Bootstrap 4
- Vue.js



Quelle: <https://www.shopware.com/de/>

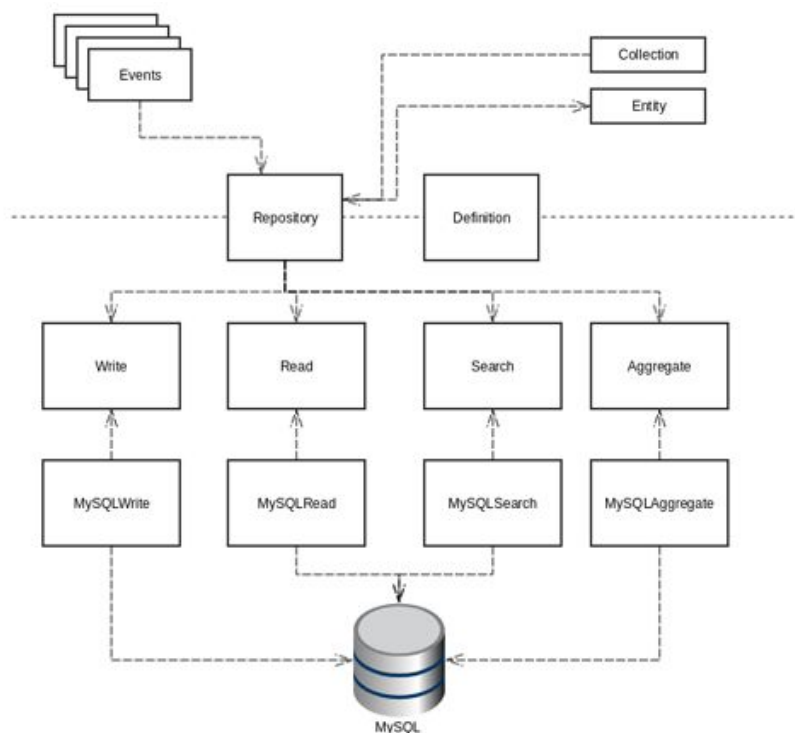
Das oben abgebildete Diagramm zeigt die Shopware 6 Plattform. Es bietet Web-Frontends für die Verwaltung verschiedener Vertriebskanäle, verfügt über eine Reihe von Benutzeroberflächen und bietet die Möglichkeit, sich über REST-APIs mit eigener Infrastruktur und externen Diensten zu verbinden.

Shopware basiert auf dem Symfony Bundle System und erweitert es mit einigen Funktionalitäten.

### 9.2. Datenbank

Der Data Abstraction Layer liefert Standardlösungen für die Übersetzbarkeit von Entities und Versionierung.

Durch Versionierung ist es möglich, mehrere Versionen einer Entität zu speichern. Alle Daten, die einer Entität untergeordnet sind, werden dupliziert und unter der neuen Version zur Verfügung gestellt. Beim Lesen oder Durchsuchen eines Datensatzes werden drei Sprachebenen durchsucht. Ein Entity stellt die Daten einer einzelnen Zeile einer Datenbanktabelle zur Verfügung. Ein Entity kann auch verknüpfte Daten enthalten.



Quelle: <https://www.webshopwerk.com/blog/shopware-data-abstraction-layer-dal/>

- ▼ Shopware6 2
  - ▶ information\_schema
  - ▼ shopware
    - ▶ tables 216
  - ▶ Server Objects

### 9.3. Teil der Tabelle – Product\_Translation

▶	product_stream_mapping
▶	product_stream_translation
▶	product_tag
▼	product_translation
	product_id binary(16)
	product_version_id binary(16)
	language_id binary(16)
	meta_description varchar(255)
	name varchar(255)
	keywords mediumtext
	description mediumtext
	meta_title varchar(255)
	pack_unit varchar(255)
	custom_fields json
	slot_config json
	created_at datetime(3)
	updated_at datetime(3)
	pack_unit_plural varchar(255)
	custom_search_keywords json
	PRIMARY (product_id, product_version_id, language_id)
	fk.product_translation.language_id (language_id) →

### 9.4. Teil der Tabelle – Product

▼	product
	id binary(16)
	version_id binary(16)
	auto_increment int (auto increment)
	product_number varchar(64)
	active tinyint unsigned
	parent_id binary(16)
	parent_version_id binary(16)
	tax_id binary(16)
	product_manufacturer_id binary(16)
	product_manufacturer_version_id binary(16)
	delivery_time_id binary(16)
	deliveryTime binary(16)
	product_media_id binary(16)
	product_media_version_id binary(16)
	cms_page_id binary(16)
	cms_page_version_id binary(16) = 0x0FA91CE3E96A4BC2...5
	unit_id binary(16)
	product_feature_set_id binary(16)
	category_tree json
	category_ids json
	option_ids json
	property_ids json
	tax binary(16)
	manufacturer binary(16)
	cover binary(16)
	unit binary(16)
	media binary(16)
	prices binary(16)
	visibilities binary(16)

## 9.5. Plugin Code

### 9.5.1. Struktur

▼	SwagMindesthaltbarkeit	
▼	src	
▼	Commands	
	SwagMindesthaltbarkeitCommand.php	16.12.21, 14:03, 7,57 kB Yesterday 18:46
▼	Resources	
▼	config	
	services.xml	16.12.21, 13:36, 2,51 kB Yesterday 13:29
▼	snippet	
▶	de_DE	
▶	en_GB	
▼	views	
▼	storefront	
▼	component	
▼	product	
▼	card	
	badges.html.twig	08.12.21, 21:34, 621 B 08.12.21, 21:36
	delivery-information.html.twig	07.12.21, 16:37, 1,39 kB 08.12.21, 21:41
▼	page	
▼	product-detail	
	buy-widget-bbd.html.twig	14.12.21, 11:10, 1,61 kB 14.12.21, 11:10
	description.html.twig	08.12.21, 20:51, 2,15 kB Yesterday 20:26
	index.html.twig	08.12.21, 20:53, 4,27 kB 14.12.21, 11:13
▼	Service	
▼	ScheduledTask	
	Cron.php	07.12.21, 13:29, 404 B 07.12.21, 13:51
	CronHandler.php	16.12.21, 14:03, 6,67 kB Yesterday 19:13
▼	SendMail	
	EmailService.php	16.12.21, 13:36, 4,02 kB Yesterday 18:56
	SalesChannelService.php	06.12.21, 21:27, 2,49 kB 08.12.21, 14:53
▼	Subscriber	
	MySubscriber.php	08.12.21, 21:32, 2,87 kB 08.12.21, 21:32
	SwagMindesthaltbarkeit.php	14.12.21, 12:10, 10,14 kB Yesterday 13:52
	composer.json	30.11.21, 00:11, 401 B 01.12.21, 15:46

### 9.5.2. SwagMindesthaltbarkeitCommand.php

```
<?php declare(strict_types=1);

namespace SwagMindesthaltbarkeit\Commands;
use Shopware\Core\Content\Mail\Service\AbstractMailService;
use Shopware\Core\Content\MailTemplate\MailTemplateEntity;
use Shopware\Core\Content\Product\ProductEntity;
use Shopware\Core\Framework\DataAbstractionLayer\
EntityRepositoryInterface;
use Shopware\Core\Framework\DataAbstractionLayer\Search\Criteria;
use Shopware\Core\Framework\DataAbstractionLayer\Search\Filter\
EqualsFilter;
use Shopware\Core\Framework\DataAbstractionLayer\Search\Filter\
RangeFilter;
use Shopware\Core\System\SalesChannel\SalesChannelContext;
use Shopware\Production\Kernel;
use SwagMindesthaltbarkeit\Service\SendMail\SalesChannelService;
use Symfony\Component\Console\Input\InputArgument;
use Symfony\Component\Console\Input\InputInterface;
use Symfony\Component\Console\Input\InputOption;
```

```

use Symfony\Component\Console\Output\OutputInterface;
use Symfony\Component\Console\Command\Command;
use SwagMindesthaltbarkeit\Service\SendMail\EmailService;
use Shopware\Core\Framework\Context;
use Symfony\Component\Mime\Email;
use Shopware\Core\Content\Mail\Service\MailService;
class SwagMindesthaltbarkeitCommand extends Command
{
    const ARG_NAME = 'argument';
    const OPT_NAME = 'option';
    protected static $defaultName = 'swag-commands:email';

    /**
     * @var EmailService
     */
    private EmailService $emailService;

    /**
     * @var SalesChannelService
     */
    private SalesChannelService $salesChannelService;

    /**
     * @var EntityRepositoryInterface
     */
    private EntityRepositoryInterface $productRepository;

    /**
     * @var EntityRepositoryInterface
     */
    private EntityRepositoryInterface $userRepository;

    /**
     * ExampleCommand constructor.
     * @param EmailService $emailService
     * @param SalesChannelService $salesChannelService
     * @param EntityRepositoryInterface $productRepository
     * @param EntityRepositoryInterface $userRepository
     * @param string|null $name
     */
    public function __construct(
        EmailService $emailService,
        SalesChannelService $salesChannelService,
        EntityRepositoryInterface $productRepository,
        EntityRepositoryInterface $userRepository,
        string $name = null
    ) {
        parent::__construct($name);
        $this->emailService = $emailService;
        $this->salesChannelService = $salesChannelService;
        $this->productRepository = $productRepository;
        $this->userRepository = $userRepository;
    }

    // Configurationen von dem Command
    protected function configure()
    {
        $this->addArgument(self::ARG_NAME, InputArgument::OPTIONAL, 'This
is an optional argument. ');
        $this->addOption(self::OPT_NAME, null,
InputOption::VALUE_OPTIONAL, 'This is an optional option. ');
    }

    /**
     * @param InputInterface $input
     * @param OutputInterface $output
     * @return int
     */

```

```

// mit der Funktion execute() wird tatsächlich die E-Mail versendet.
Alle Funktionen , die darin stehen
// werden nach dem Command durchführen ausgeführt.
protected function execute(InputInterface $input, OutputInterface
$output): int
{
    $arguments = $input->getArguments();
    $this->emailAnAdminSenden();
    return 0;
}
// sendet E-Mail an der Admin
public function emailAnAdminSenden()
{
    if (!isset($salesChannelContext)) {
        $salesChannelContext = $this->salesChannelService-
>createSalesChannelContext();
    }
    $salesChannelId=$salesChannelContext->getSalesChannel()->getId();
    $salesChannelName=$salesChannelContext->getSalesChannel()-
>getName();
    $email = $this->adminEmail();
    $mailTemplateId = $this->emailService->getMailTemplate()->getId();
    $message = $this->emailService->getMailTemplate()-
>getContentHtml();
    $message1=$this->emailService->getMailTemplate()-
>getContentPlain();
    $senderName= 'System';
    $subject=$this->emailService->getMailTemplate()->getSubject();
    $product_Tipp_List= $this->tippProducts();
    $product_List = $this->dataProducts();
    $templateData['myCustomData'] = 'Liste der Produkte die in 4 Tage
ablaufen: ';
    $templateData['myCustomData1'] = $product_List;
    $templateData['myCustomData2'] = 'Liste der Produkte die als TIPP
im Shop angezeigt
werden und in 14 Tage ablaufen:
';
    $templateData['myCustomData3'] = $product_Tipp_List;
    $this->emailService->sendMail([$email => 'Admin'], $senderName ,
$subject, $message,
    $message1, $salesChannelContext, $templateData);
}
// Ermittelt die Producte, die deaktiviert werden müssen
public function dataProducts():array
{
    $product_list=array();
    $dateTime=(new \DateTime())->add(\
DateInterval::createFromDateString('+ 4 days'));
    $dateTime1=(new \DateTime())->add(\
DateInterval::createFromDateString('+ 3 days'));
    $criteria = new Criteria();
    $criteria->addFilter(
        new RangeFilter(
            'customFields.custom_best_before_date',
            [RangeFilter::LTE => $dateTime->format(\DATE_ATOM),
            RangeFilter::GTE => $dateTime1->format(\DATE_ATOM)
        ]
    )
);
    $context = Context::createDefaultContext();

```



```

        $foundProducts = $this->productRepository->search(
            $criteria,
            $context
        )->getEntities();
        foreach ($foundProducts as $product) {
            $productNumber = $product->getProductNumber();
            array_push($product_list, $productNumber);
        }
        foreach ($foundProducts as $product) {
            $productId = $product->getId();
            $this->productRepository->update(
                [
                    ['id' => $productId, 'active' => false]
                ],
                $context
            );
        }
        return $product_list;
    }

    // Ermittelt die Liste der Produkte die als TIPP angezeigt werden
    public function tippProducts():array
    {
        $product_tipp_list=array();
        $dateTime=(new \DateTime())->add(\
DateInterval::createFromDateString('+ 14 days'));
        $dateTime1=(new \DateTime())->add(\
DateInterval::createFromDateString('+ 13 days'));
        $criteria = new Criteria();
        $criteria->addFilter(
            new RangeFilter(
                'customFields.custom_best_before_date',
                [RangeFilter::LTE => $dateTime->format(\DATE_ATOM),
                RangeFilter::GTE => $dateTime1->format(\DATE_ATOM)
            ]
        )
    );
        $context = Context::createDefaultContext();
        $foundProducts = $this->productRepository->search(
            $criteria,
            $context
        )->getEntities();
        foreach ($foundProducts as $product) {
            if($product ) {
                $productNumber = $product->getProductNumber();
                array_push($product_tipp_list, $productNumber);
            }
        }
        return $product_tipp_list;
    }

    // ermittelt die E-Mail adresse der Admin
    public function adminEmail(){
        $criteria = new Criteria();
        $criteria->addFilter(new EqualsFilter('username', 'admin'));
        $criteria->setLimit(1);
        $context = Context::createDefaultContext();
        $infoAdmin= $this->userRepository->search($criteria, $context)-
>first();
        $adminEmail=$infoAdmin->getEmail();
        return $adminEmail;
    }

```

```

    }
}

```

### 9.5.3. services.xml

```

<?xml version="1.0" ?>
<container xmlns="http://symfony.com/schema/dic/services"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://symfony.com/schema/dic/services
http://symfony.com/schema/dic/services/services-1.0.xsd">
    <services>
        <service id="SwagMindesthaltbarkeit\Service\ScheduledTask\Cron">
            <tag name="shopware.scheduled.task" />
        </service>
        <service id="SwagMindesthaltbarkeit\Service\ScheduledTask\
CronHandler" >
            <argument type="service" id="scheduled_task.repository" />
            <argument type="service" id="product.repository" />
            <argument type="service" id="SwagMindesthaltbarkeit\Service\
SendMail\EmailService"/>
            <argument type="service" id="SwagMindesthaltbarkeit\Service\
SendMail\SalesChannelService"/>
            <argument type="service" id="user.repository" />
            <tag name="messenger.message_handler" />
        </service>
        <service id="SwagMindesthaltbarkeit\Service\SendMail\EmailService"
>
            <argument type="service" id="Shopware\Core\Content\Mail\
Service\MailService"/>
            <argument type="service" id="mail_template_type.repository"/>
            <argument type="service" id="mail_template.repository"/>
            <argument type="service" id="SwagMindesthaltbarkeit\Service\
SendMail\SalesChannelService"/>
        </service>
        <service id="SwagMindesthaltbarkeit\Service\SendMail\
SalesChannelService">
            <argument type="service" id="sales_channel.repository"/>
            <argument type="service" id="Shopware\Core\System\
SalesChannel\Context\SalesChannelContextFactory" />
        </service>
        <service id="SwagMindesthaltbarkeit\Commands\
SwagMindesthaltbarkeitCommand" public="true">
            <argument type="service" id="SwagMindesthaltbarkeit\Service\
SendMail\EmailService"/>
            <argument type="service" id="SwagMindesthaltbarkeit\Service\
SendMail\SalesChannelService"/>
            <argument type="service" id="product.repository" />
            <argument type="service" id="user.repository" />
            <tag name="console.command" command="swag-commands:email" />
        </service>
        <service id="SwagMindesthaltbarkeit\Subscriber\MySubscriber">
            <argument type="service" id="logger" />
            <argument type="service" id="product.repository" />
            <tag name="kernel.event_subscriber" />
        </service>
    </services>
</container>

```

#### 9.5.4. swag\_mindesthaltbarkeit.de\_De.json

```
{
  "customFields": {
    "custom_best_before_date": "Mindesthaltbarkeitsdatum: ",
    "custom_best_before_after_delivery": "Mindesthaltbarkeit nach der
Lieferung : "
  }
}
```

#### 9.5.5. swag\_mindesthaltbarkeit.en\_GB.json

```
{
  "customFields": {
    "custom_best_before_date": "Best before : ",
    "custom_best_before_after_delivery": "Best before after delivery : "
  }
}
```

#### 9.5.6. badges.html.twig

```
{% sw_extends '@Storefront/storefront/component/product/card/badges.html.twig' %}

{% block component_product_badges_topseller %}
  {% if product.translated.customFields.custom_best_before_date|
date('U') <= ("now"|date('U') + 864000 )
    and product.translated.customFields.custom_best_before_date|
date('U') >= ("now"|date('U') + 345600 )%}
    <div>
      <span class="badge badge-warning badge-topseller"> {{
"listing.boxLabelTopseller"|trans|sw_sanitize }} </span>
    </div>
  {% else %}
    {{ parent() }}
  {% endif %}
{% endblock %}
```

#### 9.5.7. delivery-information.html.twig

```
{% sw_extends '@Storefront/storefront/component/delivery-
information.html.twig' %}

{% block component_delivery_information %}
  {% if page is defined and page.product is defined %}
    {% set product = page.product %}
  {% endif %}
  <div class="product-delivery-information">
    {% block component_best_before_information %}
      {% if
page.product.translated.customFields.custom_best_before_date|date('U') >
("now"|date('U') + 345600)
        and (product.releaseDate|date('U') < "now"|date('U') +
345600)
          and not general.deliveryNotAvailable
          and not
page.product.translated.customFields.custom_best_before_after_delivery %}
        <p class="delivery-information date-info">
          <span class="delivery-status-indicator bg-
info"></span>
```

```

        <strong>{{ "customFields.custom_best_before_date" |
trans }}</strong>

{{ page.product.translated.customFields.custom_best_before_date |
format_date('long', locale=app.request.locale)}}
    </p>
    {% elseif
page.product.translated.customFields.custom_best_before_date|date('U') <=
("now"|date('U') + 345600) %}
        <p class="delivery-information date-info"></p>
        {% endif %}
    {% endblock %}
    {{ parent() }}
</div>
{% endblock %}

```

### 9.5.8. buy-widjet-bbd.html.twig

```

{% sw_extends '@Storefront/storefront/component/delivery-
information.html.twig' %}

{% block component_delivery_information %}
    {% if page is defined and page.product is defined %}
        {% set product = page.product %}
    {% endif %}
    <div class="product-delivery-information">
        {% block component_best_before_information %}
            {% if
page.product.translated.customFields.custom_best_before_date|date('U') >
("now"|date('U') + 345600)
                and (product.releaseDate|date('U') < "now"|date('U')+
345600)
                and not general.deliveryNotAvailable
                and not
page.product.translated.customFields.custom_best_before_after_delivery %}
                <p class="delivery-information date-info">
                    <span class="delivery-status-indicator bg-
info"></span>
                    <strong>{{ "customFields.custom_best_before_date" |
trans }}</strong>

                    {{ page.product.translated.customFields.custom_best_before_date |
format_date('long', locale=app.request.locale)}}
                </p>
            {% elseif
page.product.translated.customFields.custom_best_before_date|date('U') <=
("now"|date('U') + 345600) %}
                <p class="delivery-information date-info"></p>
            {% endif %}
        {% endblock %}
        {{ parent() }}
    </div>
{% endblock %}

```

### 9.5.9. description.html.twig

```

{% sw_extends
'@Storefront/storefront/page/product-detail/description.html.twig'
%}

{% block utilities_offcanvas_content %}
    {% block page_product_detail_description_container %}

```

```

<div class="product-detail-description tab-pane-container">
    {% block page_product_detail_description_title %}
        <div class="h3 product-detail-description-title">
            {{ "detail.descriptionTitle"|trans|sw_sanitize }}
        </div>
    {% endblock %}
    {% block page_product_detail_description_content %}
        {% block
page_product_detail_description_content_mhd_after_delivery %}
            <div class="product-detail-description-mhd"
                itemprop="description">
                {% if
page.product.translated.customFields.custom_best_before_after_delivery
                    and not general.deliveryNotAvailable%}
                    <p>
                        <strong>{{
"customFields.custom_best_before_after_delivery"|trans|
sw_sanitize }}</strong>
                    {{ page.product.translated.customFields.custom_best_before_after_delivery
                    }}
                        </p>
                    {% else %}
                        <p></p>
                    {% endif %}
                </div>
            {% endblock %}
            {% block page_product_detail_description_content_text %}
                <div class="product-detail-description-text"
                    itemprop="description">
                    {{ page.product.translated.description|raw }}
                </div>
            {% endblock %}
        {% block
page_product_detail_description_content_properties %}
            {% if page.product.sortedProperties|length > 0 %}
                {% sw_include
"@Storefront/storefront/page/product-detail/properties.html.twig" %}
            {% endif %}
        {% endblock %}
    {% endblock %}
</div>
{% endblock %}

```

### 9.5.10. index.html.twig

```

{% sw_extends
"@Storefront/storefront/page/product-detail/index.html.twig" %}
{% block base_content %}
    {% block page_product_detail %}
        <div class="product-detail"
            itemscope
            itemtype="https://schema.org/Product">
            {% block page_product_detail_inner %}
                {% block page_product_detail_content %}
                    <div class="product-detail-content">
                        {% block page_product_detail_headline %}

```

```

<div class="row align-items-center product-
detail-headline">
    {% sw_include
'@Storefront/storefront/page/product-detail/headline.html.twig' %}
    </div>
{% endblock %}
{% set mediaItems = page.product.media.media %}
{% block page_product_detail_main %}
    <div class="row product-detail-main">
        {% block page_product_detail_media %}
            <div class="col-lg-7 product-detail-
media">
                {% if page.product.media %}
                    {% sw_include
'@Storefront/storefront/element/cms-element-image-gallery.html.twig' with
{
                        'mediaItems': mediaItems,
                        'zoom': true,
                        'zoomModal': true,
                        'displayMode': 'contain',
                        'gutter': 5,
                        'minHeight': '430px',
                        'navigationArrows':
'inside',
                        'navigationDots':
'inside',
                        'galleryPosition': 'left',
                        'isProduct': true,
                        'fallbackImageTitle':
page.product.translated.name,
                        'startIndexThumbnails':
page.product.cover.position + 1,
                        'startIndexSlider':
page.product.cover.position + 1
                    } %}
                {% endif %}
            </div>
        {% endblock %}
        {% block page_product_detail_buy %}
            <div class="col-lg-5 product-detail-
buy">
                {% if
page.product.translated.customFields.custom_best_before_date|date('U') <=
("now"|date('U') + 345600) and
page.product.translated.customFields.custom_best_before_check == true
                %}
                    {% sw_include
'@Storefront/storefront/page/product-detail/buy-widget-bbd.html.twig' %}
                {% else %}
                    {% sw_include
'@Storefront/storefront/page/product-detail/buy-widget.html.twig' %}
                {% endif %}
            </div>
        {% endblock %}
    </div>
{% endblock %}
{% block page_product_detail_tabs %}
    <div class="product-detail-tabs">

```

```

                {% sw_include
'@Storefront/storefront/page/product-detail/tabs.html.twig' %}
            </div>
        {% endblock %}
        {% block page_product_detail_cross_selling %}
            <div class="product-detail-tabs product-detail-cross-
selling">
                {% sw_include
'@Storefront/storefront/page/product-detail/cross-selling/tabs.html.twig'
with {
                    crossSellings: page.crossSellings
                } %}
            </div>
        {% endblock %}
    {% endblock %}
</div>
{% endblock %}

```

### 9.5.11. Cron.php

<?php

```

declare(strict_types=1);
namespace SwagMindesthaltbarkeit\Service\ScheduledTask;
use Shopware\Core\Framework\MessageQueue\ScheduledTask\ScheduledTask;
class Cron extends ScheduledTask
{
    public static function getTaskName(): string
    {
        return 'swag.cron_task';
    }
    public static function getDefaultInterval(): int
    {
        return 300; // 24 Stunden in Sekunden
    }
}

```

### 9.5.12. CronHandler.php

<?php

```

declare(strict_types=1);
namespace SwagMindesthaltbarkeit\Service\ScheduledTask;
use Shopware\Core\Framework\Context;
use Shopware\Core\Framework\DataAbstractionLayer\EntityRepositoryInterface;
use Shopware\Core\Framework\DataAbstractionLayer\Search\Criteria;
use Shopware\Core\Framework\DataAbstractionLayer\Search\Filter\EqualsFilter;
use Shopware\Core\Framework\DataAbstractionLayer\Search\Filter\RangeFilter;
use Shopware\Core\Framework\MessageQueue\ScheduledTask\ScheduledTaskHandler;
use SwagMindesthaltbarkeit\Service\SendMail\EmailService;
use SwagMindesthaltbarkeit\Service\SendMail\SalesChannelService;
class CronHandler extends ScheduledTaskHandler
{
    /**
     * @var EmailService
     */
    private $emailService;
}

```

```

/**
 * @var EntityRepositoryInterface
 */
private $productRepository;
/**
 * @var SalesChannelService
 */
/**
 * @var EntityRepositoryInterface
 */
private EntityRepositoryInterface $userRepository;
private $salesChannelService;
public function __construct(
    EntityRepositoryInterface $scheduledTaskRepository,
    EntityRepositoryInterface $product_repo,
    EmailService $emailService,
    SalesChannelService $salesChannelService,
    EntityRepositoryInterface $userRepository
) {
    parent::__construct($scheduledTaskRepository);
    $this->productRepository = $product_repo;
    $this->emailService = $emailService;
    $this->scheduledTaskRepository = $scheduledTaskRepository;
    $this->salesChannelService = $salesChannelService;
    $this->userRepository = $userRepository;
}
/**
 *
 * @return iterable
 */
public static function getHandledMessages(): iterable
{
    return [ Cron::class ];
}
public function run(): void
{
    $this->emailAnAdminSenden();
}
public function emailAnAdminSenden()
{
    // wenn kein salesChannelContext , create
    if (!isset($salesChannelContext)) {
        $salesChannelContext = $this->salesChannelService->createSalesChannelContext();
    }
    $salesChannelId=$salesChannelContext->getSalesChannel()->getId();
    $salesChannelName=$salesChannelContext->getSalesChannel()->getName();
    $email = $this->adminEmail();
    $mailTemplateId = $this->emailService->getMailTemplate()->getId();
    $message=$this->emailService->getMailTemplate()->getContentHtml();
    $message1=$this->emailService->getMailTemplate()->getContentPlain();
    $senderName= 'System';
    $subject=$this->emailService->getMailTemplate()->getSubject();
    //Liste der Produkte , die in 14 Tagen ablaufen
    $product_Tipp_List= $this->tippProducts();
    //Liste der Produkte , die in 4 Tagen ablaufen
    $product_List = $this->dataProducts();

```



```
// Die eigene Data , die in Template angezeigt wird, und veränderbar ist.
$templateData['myCustomData'] = 'Liste der Produkte die in 4 Tage
ablaufen: ';
$templateData['myCustomData1'] = $product_List;
$templateData['myCustomData2'] = 'Liste der Produkte die als TIPP
im Shop angezeigt
werden und in 14 Tagen ablaufen:
';
$templateData['myCustomData3'] = $product_Tipp_List;
// sende Email an Admin
$this->emailService->sendMail([$email => 'Admin'], $senderName ,
$subject, $message,
    $message1, $salesChannelContext, $templateData);
}
// Funktion, die die Liste in 4 Tagen ablaufende und deaktivierte
Produkte zurückgibt
public function dataProducts():array
{
    $product_list=array();
    $dateTime=(new \DateTime())->add(\
DateInterval::createFromDateString('+4 days'));
    $dateTime1=(new \DateTime())->add(\
DateInterval::createFromDateString('+3 days'));
    $criteria = new Criteria();
    $criteria->addFilter(
        new RangeFilter(
            'customFields.custom_best_before_date',
            [RangeFilter::LTE => $dateTime->format(\DATE_ATOM),
            RangeFilter::GTE => $dateTime1->format(\DATE_ATOM)
            ]
        )
    );
    $context = Context::createDefaultContext();
    $foundProducts = $this->productRepository->search(
        $criteria,
        $context
    )->getEntities();
    // Der Produktnummer wird in der Liste zugefügt
    foreach ($foundProducts as $product) {
        if($product ) {
            $productNumber = $product->getProductNumber();
            array_push($product_list, $productNumber);
        }
    }
    //Die Produkte aus der Liste werden upgedated und deaktiviert
    foreach ($foundProducts as $product) {
        if($product ) {
            $productId = $product->getId();
            $this->productRepository->update(
                [
                    ['id' => $productId, 'active' => false]
                ],
                $context
            );
        }
    }
    return $product_list;
}
// Funktion, die die Liste in 14 Tagen ablaufende Produkte zurückgibt
```

```

public function tippProducts():array
{
    $product_tipp_list=array();
    $dateTime=(new \DateTime())->add(\
DateInterval::createFromDateString('+ 14 days'));
    $dateTime1=(new \DateTime())->add(\
DateInterval::createFromDateString('+13 days'));
    $criteria = new Criteria();
    $criteria->addFilter(
        new RangeFilter(
            'customFields.custom_best_before_date',
            [RangeFilter::LTE => $dateTime->format(\DATE_ATOM),
            RangeFilter::GTE => $dateTime1->format(\DATE_ATOM)
        ]
    );
    $context = Context::createDefaultContext();
    $foundProducts = $this->productRepository->search(
        $criteria,
        $context
    )->getEntities();
    foreach ($foundProducts as $product) {
        if($product ){
            $productNumber = $product->getProductNumber();
            array_push($product_tipp_list, $productNumber);
        }
    }
    return $product_tipp_list;
}

// ermittelt die E-Mail adresse der Admin
public function adminEmail(){
    $criteria = new Criteria();
    $criteria->addFilter(new EqualsFilter('username', 'admin'));
    $criteria->setLimit(1);
    $context = Context::createDefaultContext();
    $infoAdmin= $this->userRepository->search($criteria, $context)-
>first();
    $adminEmail=$infoAdmin->getEmail();
    return $adminEmail;
}
}

```

### 9.5.13. EmailService.php

<?php

```

namespace SwagMindesthaltbarkeit\Service\SendMail;
use Shopware\Core\Content\Mail\Service\AbstractMailService;
use Shopware\Core\Content\MailTemplate\MailTemplateEntity;
use Shopware\Core\Framework\Context;
use Shopware\Core\Framework\DataAbstractionLayer\
EntityRepositoryInterface;
use Shopware\Core\Framework\DataAbstractionLayer\Search\Criteria;
use Shopware\Core\Framework\DataAbstractionLayer\Search\Filter\
EqualsFilter;
use Shopware\Core\Framework\Validation\DataBag\DataBag;
use Shopware\Core\System\SalesChannel\SalesChannelContext;
use Shopware\Core\Content\Mail\Service\MailService;
use SwagMindesthaltbarkeit\Service\SendMail\SalesChannelService;
class EmailService
{

```

```

private $mailService;
private $mailTemplateTypeRepository;
private $mailTemplateRepository;
private $salesChannelService;
public function __construct(
    AbstractMailService $mailService,
    EntityRepositoryInterface $mailTemplateTypeRepository,
    EntityRepositoryInterface $mailTemplateRepository,
    SalesChannelService $salesChannelService
)
{
    $this->mailService = $mailService;
    $this->mailTemplateTypeRepository = $mailTemplateTypeRepository;
    $this->mailTemplateRepository = $mailTemplateRepository;
    $this->salesChannelService = $salesChannelService;
}
/**
 * @param array $recipients
 * @param string $senderName
 * @param string $subject
 * @param string $messageHtml
 * @param string $messagePlain
 * @param SalesChannelContext|null $salesChannelContext
 * @param array $templateData
 */
public function sendMail(
    array $recipients,
    string $senderName,
    string $subject,
    string $messageHtml ,
    string $messagePlain ,
    SalesChannelContext $salesChannelContext ,
    array $templateData = []
) : void
{
    $data = new DataBag();
    $mailTemplate = $this->getMailTemplate();
    $data->set('templateId', $mailTemplate->getId());
    //basic e-mail data
    $data->set('recipients', $recipients); //format: ['email
address' => 'recipient name']
    $data->set('senderName', $senderName);
    $data->set('subject', $subject);
    $data->set('contentHtml', $mailTemplate->getContentHtml());
    $data->set('contentPlain', $mailTemplate->getContentPlain());
    $data->set('templateData', $templateData); //format:
['myCustomData' => 'Example']
    // create SelsChannelContext falls noch nicht gibt
    if (!isset($salesChannelContext)) {
        $salesChannelContext = $this->salesChannelService-
>createSalesChannelContext();
    }
    $data->set('salesChannelId', $salesChannelContext-
>getSalesChannel()->getId());
    //send die e-mail
    $this->mailService->send($data->all(), $salesChannelContext-
>getContext(), $templateData);
}
/**
 * @param string $id

```

```

    * @param Context $context
    * @return MailTemplateEntity|null
    */
    public function getMailTemplate(string $technicalName =
'admin_custom_mail_type', Context $context = null): ?MailTemplateEntity
    {
        //gibt SalesChannelContext falls noch nicht present
        if (!isset($context)) {
            $salesChannelContext = $this->salesChannelService-
>createSalesChannelContext();
            $context = $salesChannelContext->getContext();
        }
        $criteria = new Criteria();
        $criteria->addFilter(new EqualsFilter('technicalName',
$technicalName));
        $templateType=$this->mailTemplateTypeRepository->search($criteria,
$context)->first();
        $tempTypeId=$templateType->getId();
        // setzt die Kriterien nach dem wir suchen in den mail template
repository
        $criteria = new Criteria();
        $criteria->addFilter(new EqualsFilter('mailTemplateTypeId',
$TempTypeId));
        $criteria->setLimit(1);
        //get and return one template
        return $this->mailTemplateRepository->search($criteria, $context)-
>first();
    }
}

```

### 9.5.14. SalesChannelService.php

<?php

```

namespace SwagMindesthaltbarkeit\Service\SendMail;
use Shopware\Core\Framework\Context;
use Shopware\Core\Framework\DataAbstractionLayer\
EntityRepositoryInterface;
use Shopware\Core\Framework\DataAbstractionLayer\Search\Criteria;
use Shopware\Core\Framework\DataAbstractionLayer\Search\Filter\
EqualsFilter;
use Shopware\Core\System\SalesChannel\Context\
AbstractSalesChannelContextFactory;
use Shopware\Core\System\SalesChannel\Context\SalesChannelContextService;
use Shopware\Core\System\SalesChannel\SalesChannelContext;
use Shopware\Core\System\SalesChannel\SalesChannelEntity;
class SalesChannelService
{
    private $salesChannelRepository;
    private $salesChannelContextFactory;
    public function __construct(
        EntityRepositoryInterface $salesChannelRepository,
        AbstractSalesChannelContextFactory $salesChannelContextFactory
    )
    {
        $this->salesChannelRepository = $salesChannelRepository;
        $this->salesChannelContextFactory = $salesChannelContextFactory;
    }
    /**
     * @param string|null $salesChannelId
     * @return SalesChannelContext

```

```

        */
        public function createSalesChannelContext(string $salesChannelId =
null) : SalesChannelContext
        {
            if (!isset($salesChannelId) || !isset($languageId)) {
                $criteria = new Criteria();
                if (isset($salesChannelId)) {
                    $criteria->addFilter(new EqualsFilter('salesChannelId',
$salesChannelId));
                }
                /** @var SalesChannelEntity $salesChannel */
                $salesChannel = $this->salesChannelRepository-
>search($criteria, Context::createDefaultContext())->first();
                if ($salesChannel) {
                    $salesChannelId = $salesChannel->getId();
                }
            }
            return $this->salesChannelContextFactory->create('',
$salesChannelId);
        }
    }
}

```

#### 9.5.15. MySubscriber.php

```

<?php declare(strict_types=1);

namespace SwagMindesthaltbarkeit\Subscriber;
use Shopware\Core\Framework\DataAbstractionLayer\Search\Criteria;
use Shopware\Core\Framework\DataAbstractionLayer\
EntityRepositoryInterface;
use Shopware\Core\Framework\DataAbstractionLayer\Event\EntityLoadedEvent;
use Psr\Log\LoggerInterface;
use Shopware\Core\Framework\DataAbstractionLayer\Search\Filter\
MultiFilter;
use Shopware\Core\Framework\DataAbstractionLayer\Search\Filter\
RangeFilter;
use Shopware\Core\Framework\DataAbstractionLayer\Search\Filter\
EqualsFilter;
use Shopware\Core\Framework\Context;
use Symfony\Component\EventDispatcher\EventSubscriberInterface;
use Shopware\Core\Content\Product\ProductEvents;
use Shopware\Core\Content\Cms\CmsPageEvents;
class MySubscriber implements EventSubscriberInterface
{
    /**
     * @var LoggerInterface;
     */
    private $logger;
    /**
     * @var EntityRepositoryInterface
     */
    private EntityRepositoryInterface $productRepository;
    public function __construct(LoggerInterface $logger,
EntityRepositoryInterface $productRepository)
    {
        $this->logger = $logger;
        $this->productRepository = $productRepository;
    }
    public static function getSubscribedEvents(): array
    {
        return [

```

```

        CmsPageEvents::PAGE_LOADED_EVENT => 'onNavigationPageLoaded'
    ];
}
public function onNavigationPageLoaded(EntityLoadedEvent
$entityLoadedEvent)
{
    $dateTime = (new \DateTime())->add(\
DateInterval::createFromDateString('+4 days'));
    $criteria = new Criteria();
    $criteria->addFilter(
        new RangeFilter(
            'customFields.custom_best_before_date',
            [
                RangeFilter::LTE => $dateTime->format(\DATE_ATOM),
            ]
        )
    );
    $criteria->addFilter(new
EqualsFilter('customFields.custom_best_before_check', true));
    $context = Context::createDefaultContext();
    $foundProducts = $this->productRepository->search(
        $criteria,
        $context
    )->getEntities();
    foreach ($foundProducts as $product) {
        $productId = $product->getId();
        $this->productRepository->update(
            [
                ['id' => $productId, 'active' => true],
            ],
            $context
        );
    }
}
}

```

### 9.5.16. SwagMindesthaltbarkeit.php

```

<?php declare(strict_types=1);

namespace SwagMindesthaltbarkeit;
use Doctrine\DBAL\Exception\UniqueConstraintViolationException;
use Shopware\Core\Content\MailTemplate\Aggregate\MailTemplateType\
MailTemplateTypeEntity;
use Shopware\Core\Framework\DataAbstractionLayer\
EntityRepositoryInterface;
use Shopware\Core\Framework\DataAbstractionLayer\Search\Criteria;
use Shopware\Core\Framework\DataAbstractionLayer\Search\Filter\
EqualsFilter;
use Shopware\Core\Framework\Plugin;
use Shopware\Core\Framework\Plugin\Context\InstallContext;
use Shopware\Core\Framework\Plugin\Context\UninstallContext;
use Shopware\Core\Framework\Uuid\Uuid;
use Shopware\Core\System\CustomField\CustomFieldTypes;
class SwagMindesthaltbarkeit extends Plugin
{
    public const TEMPLATE_TYPE_NAME = 'List of expired products';
    public const TEMPLATE_TYPE_TECHNICAL_NAME =
'admin_custom_mail_type';
    public const MAIL_TEMPLATE_NAME = 'MyAdminMailTemplate';
}

```

```

public function install(InstallContext $installContext): void
{
    $this->zusatzfelderSet();

    /** @var EntityRepositoryInterface
    $mailTemplateTypeRepository */
    $mailTemplateTypeRepository = $this->container-
>get('mail_template_type.repository');
    /** @var EntityRepositoryInterface $mailTemplateRepository
    */
    $mailTemplateRepository = $this->container-
>get('mail_template.repository');
    $mailTemplateTypeId = Uuid::randomHex();
    $mailTemplateType = [
        [
            'id' => $mailTemplateTypeId,
            'name' => self::TEMPLATE_TYPE_NAME,
            'technicalName' =>
self::TEMPLATE_TYPE_TECHNICAL_NAME,
            'availableEntities' => [
                'product' => 'product',
                'salesChannel' => 'sales_channel'
            ]
        ]
    ];
    $mailTemplate = [
        [
            'id' => Uuid::randomHex(),
            'mailTemplateTypeId' => $mailTemplateTypeId,
            'subject' => [
                'en-GB' => 'Soon to be expired products ',
                'de-DE' => 'Bald ablaufende Produkte'
            ],
            'contentPlain' => "{{ myCustomData }}"
                                {% for product in myCustomData1
                                {{ product }}
                                {% endfor %}
                                {{ myCustomData2 }}
                                {% for product in myCustomData3
                                {{ product }}
                                {% endfor %}}",
            'contentHtml' => '<div style="font-family:sans-
serif; font-size:12px;">
                                <p>{{ myCustomData }}</p>
                                <ol>
                                    {% for product in
myCustomData1 %}
<li>{{ product }}</li>
                                    {% endfor %}
                                </ol>
                                <p>{{ myCustomData2 }}</p>
                                <ol>
                                    {% for product in
myCustomData3 %}
<li>{{ product }}</li>
                                    {% endfor %}

```

```

        </ol>
    </div>',

    ];
    try {
        $mailTemplateTypeRepository->create($mailTemplateType,
$installContext->getContext());
        $mailTemplateRepository->create($mailTemplate,
$installContext->getContext());
    } catch (UniqueConstraintViolationException $exception) {
    }
}

public function uninstall(UninstallContext $uninstallContext):
void
{
    if ($uninstallContext->keepUserData()) {
        return;
    }
    /** @var EntityRepositoryInterface
$mailTemplateTypeRepository */
    $mailTemplateTypeRepository = $this->container-
>get('mail_template_type.repository');
    /** @var EntityRepositoryInterface $mailTemplateRepository
*/
    $mailTemplateRepository = $this->container-
>get('mail_template.repository');
    /** @var MailTemplateTypeEntity $myAdminMailTemplateType */
    $myAdminMailTemplateType = $mailTemplateTypeRepository-
>search(
        (new Criteria())
            ->addFilter(new EqualsFilter('technicalName',
self::TEMPLATE_TYPE_TECHNICAL_NAME)),
        $uninstallContext
            ->getContext()
    )->first();
    $mailTemplateIds = $mailTemplateRepository->searchIds(
        (new Criteria())
            ->addFilter(new EqualsFilter('mailTemplateTypeId',
$myAdminMailTemplateType->getId())),
        $uninstallContext
            ->getContext()
    )->getIds();
    $ids = array_map(
        static function ($id) {
            return ['id' => $id];
        },
        $mailTemplateIds
    );
    $mailTemplateRepository->delete($ids, $uninstallContext-
>getContext());
    //Delete the TemplateType which were added by this Plugin
    $mailTemplateTypeRepository->delete(
        [
            ['id' => $myAdminMailTemplateType->getId()]
        ],
        $uninstallContext->getContext()
    );
}

public function zusatzfelderSet(InstallContext $installContext): void
{

```



```

/** @var EntityRepositoryInterface $customFieldsSetRepository */

$customFieldSetRepository = $this->container-
>get('custom_field_set.repository');
$customFieldSet=
[
    [
        'id'=> Uuid::randomHex(),
        'name' => 'custom_best_before',
        'config' => [
            'label' => [
                'en-GB' => 'Best before',
                'de-DE' => 'Mindesthaltbarkeit',
            ],
        ],
        'relations' => [
            [
                'id'=> Uuid::randomHex(),
                'entityName' => 'product',
            ]
        ],
        'customFields' => [
            [
                'id'=> Uuid::randomHex(),
                'name' => 'custom_best_before_date',
                'type' => CustomFieldTypes::DATETIME,
                'config' => [
                    'label' => [
                        'type' => 'date',
                        'de-DE' =>
'Mindesthaltbarkeitsdatum:',
                        'en-GB' => 'Best befor :',
                    ],
                    'config' => [
                        'time_24hr' => true,
                        'dateType' => 'datetime'
                    ],
                    'componentName'=> 'sw-field',
                    'customFieldType' => 'date',
                    'customFieldPosition' => 1
                ],
            ],
        ],
        [
            'id'=> Uuid::randomHex(),
            'name' => 'custom_best_before_check',
            'type' => CustomFieldTypes::SWITCH,
            'config' => [
                'type' => 'switch',
                'label' => [
                    'de-DE' => 'Das Produkt wird nach
ablaufen des MHD weiter im Shop
zu sehen sein aber nicht
verkauft werden',
                    'en-GB' => 'The product will continue
to be seen in the shop after
the best-before date has
expired, but will not be sold'
                ],
                'componentName'=> 'sw-field',
                'customFieldType' => 'switch',
            ],
        ],
    ],
]

```

```

        'customFieldPosition' => 2
    ],
    [
        'id'=> Uuid::randomHex(),
        'name' => 'custom_best_before_check',
        'type' => CustomFieldTypes::TEXT,
        'config' => [
            'type' => 'text',
            'label' => [
                'de-DE' => 'Mindesthaltbar nach der
Lieferung :',
                'en-GB' => 'Best befor after
delivery:'
            ],
            'componentName'=> 'sw-field',
            'customFieldType' => 'text',
            'customFieldPosition' => 3
        ]
    ],
],
];
try {
    $customFieldSetRepository->create($customFieldSet, $installContext->getContext());
} catch (UniqueConstraintViolationException
$exception) {
}
}
}

```

### 9.5.17. composer.json

```

{
    "name": "swag/plugin-skeleton",
    "description": "Skeleton plugin",
    "type": "shopware-platform-plugin",
    "license": "MIT",
    "autoload": {
        "psr-4": {
            "SwagMindesthaltbarkeit\\": "src/"
        }
    },
    "extra": {
        "shopware-plugin-class": "SwagMindesthaltbarkeit\\
SwagMindesthaltbarkeit",
        "label": {
            "de-DE": "Mindesthaltbarkeit",
            "en-GB": "BestBeforeDate"
        }
    }
}

```

## 9.6. Administration – Meine Erweiterungen

**Administration**  
v6.4.3.1 Stable Version

**Meine Erweiterungen**

Erweiterungen Durchsuche alle Erweiterungen ...

Erweiterung hochladen

Apps Themes Empfehlungen Shopware Account

☐ Inaktive Erweiterungen ausblenden

Zuletzt aktualisiert

| Erweiterung                              | Status      | Version        | Installiert am | Aktionen                         |
|------------------------------------------|-------------|----------------|----------------|----------------------------------|
| PayPal-Produkte für Shopware 6 (inaktiv) | Inaktiv     | Version: 3.3.1 | 10.09.2021     | <a href="#">Aktualisierung</a>   |
| Shopware 6 Demodaten                     | Installiert | Version: 1.0.8 |                | <a href="#">App installieren</a> |
| Mindesthaltbarkeit                       | Installiert | Version: 1.0.0 | 30.11.2021     |                                  |

## 9.7. Administration – Mindesthaltbarkeit ZusatzfeldSet mit 2 Felder

**Administration**  
v6.4.3.1 Stable Version

**Mindesthaltbarkeit**

Verwenden für: Produkte

Suche ...

| Label                              | Typ               | Position |
|------------------------------------|-------------------|----------|
| Mindesthaltbarkeitsdatum           | Datum- / Zeitfeld | 1        |
| Mindesthaltbar nach der Lieferung: | Textfeld          | 2        |

Abbrechen Speichern

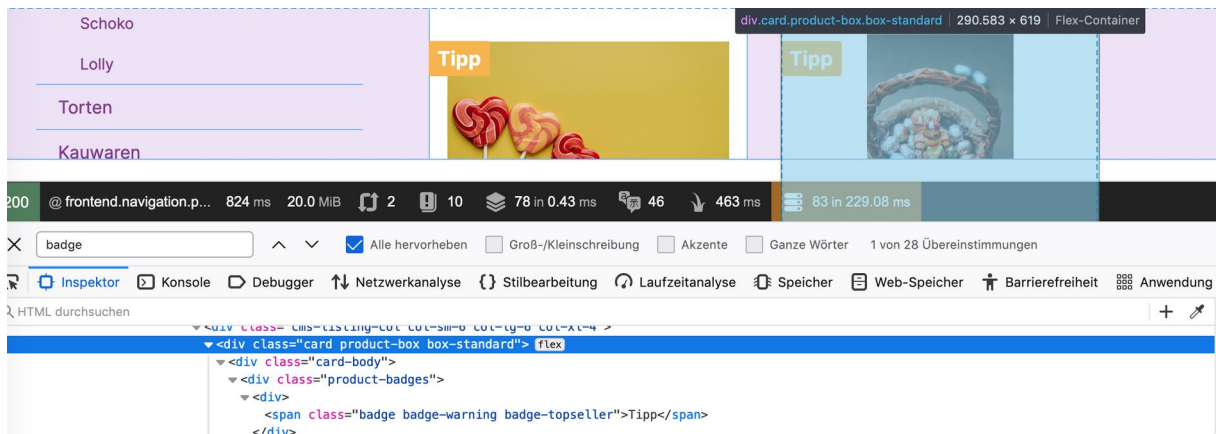
## 9.8. Suche nach Informationen

@Storefront/storefront/page/product-detail/buy-widget-price.html.twig

@Storefront/storefront/component/delivery-information.html.twig

@Storefront/storefront/page/product-detail/buy-widget-form.html.twig

## Shopware 6 Erweiterung für Online-Shop



## 9.9.Profiler

**Symfony Profiler** search on symfony.com Search

https://praktikant.vomwege.gmbh/detail/1abcb968d7bd4bf49eac9bed7adff962

Method: GET HTTP Status: 200 IP: 77.1.182.89 Profiled on: Wed, 01 Dec 2021 14:43:21 +0100 Token: c40c24

Last 10 Latest Search

**Request / Response**

**Performance**

Validator

Exception

Logs 7

Events

Routing

Cache

Translation

**Twig**

Debug

Doctrine

Messages

E-mails

Message Queue

Configuration

Settings

**Twig Metrics**

|             |                |             |             |
|-------------|----------------|-------------|-------------|
| 466 ms      | 111            | 464         | 0           |
| Render time | Template calls | Block calls | Macro calls |

Render time includes sub-requests rendering time (if any).

**Rendered Templates**

| Template Name & Path                                                                                                                                    | Render Count |
|---------------------------------------------------------------------------------------------------------------------------------------------------------|--------------|
| @Storefront/storefront/page/product-detail/index.html.twig<br>vendor/shopware/storefront/Resources/views/storefront/page/product-detail/index.html.twig | 1            |
| @Storefront/storefront/base.html.twig<br>vendor/shopware/storefront/Resources/views/storefront/base.html.twig                                           | 1            |
| @Storefront/storefront/page/product-detail/meta.html.twig<br>vendor/shopware/storefront/Resources/views/storefront/page/product-detail/meta.html.twig   | 1            |
| @Storefront/storefront/layout/meta.html.twig<br>vendor/shopware/storefront/Resources/views/storefront/layout/meta.html.twig                             | 1            |
| @Storefront/storefront/component/feature.html.twig<br>vendor/shopware/storefront/Resources/views/storefront/component/feature.html.twig                 | 1            |
| @Storefront/storefront/component/analytics.html.twig<br>vendor/shopware/storefront/Resources/views/storefront/component/analytics.html.twig             | 1            |
| @Storefront/storefront/component/recaptcha.html.twig<br>vendor/shopware/storefront/Resources/views/storefront/component/recaptcha.html.twig             | 1            |
| @Storefront/storefront/utilities/alert.html.twig<br>vendor/shopware/storefront/Resources/views/storefront/utilities/alert.html.twig                     | 2            |
| @Storefront/storefront/utilities/icon.html.twig<br>vendor/shopware/storefront/Resources/views/storefront/utilities/icon.html.twig                       | 38           |

## Shopware 6 Erweiterung für Online-Shop

Symfony Exception

ErrorException > RuntimeException

HTTP 500 Internal Server Error

An exception has been thrown during the rendering of a template ("Warning: A non-numeric value encountered").

Exceptions 2 Logs 1 Stack Traces 2

Twig\Error\RuntimeException

in custom/plugins/SwagMindesthaltbarkeit/src/Resources/views/storefront/component/delivery-information.html.twig (line 12)

```

7.
8.     <div class="product-delivery-information">
9.
10.         {% block component_best_before_information %}
11.             {% if page.product.translated.customFields.custom_best_before_date|date('U') > ("now"|date('U') + 345600)
12.                 and (product.releaseDate|date('U') < "now"|date('U') + 345600))
13.                 and not general.deliveryNotAvailable
14.                 and not page.product.translated.customFields.custom_best_before_after_delivery %}
15.
16.                 <p class="delivery-information date-info">
17.                     <span class="delivery-status-indicator bg-info"></span>

```

in vendor/twig/twig/src/Environment.php(358) : eval()'d code -> displayBlock (line 83)

in vendor/twig/twig/src/Template.php -> block\_component\_delivery\_information (line 171)

in vendor/twig/twig/src/Environment.php(358) : eval()'d code -> displayBlock (line 50)

### 9.10. Scheduled-Task in der Datenbank

Bearbeiten: scheduled\_task

|                      |                                                   |
|----------------------|---------------------------------------------------|
| id                   | DE50DD7374F74411B8029D138101D375                  |
| name                 | swag.cron_task                                    |
| scheduled_task_class | SwagMindesthaltbarkeit\Service\ScheduledTask\Cron |
| run_interval         | 86400                                             |
| status               | scheduled                                         |
| last_execution_time  | 2021-12-07 12:12:00.839                           |
| next_execution_time  | 2021-12-07 12:15:00.836                           |
| created_at           | 2021-12-07 12:12:00.837                           |
| updated_at           | NULL                                              |

Speichern Speichern und weiter bearbeiten Entfernen

### 9.11. E-Mail , die der Admin bekommt

galani@vomwege-it.de

Auswahl Konversations... Optionen Aktualisieren

Antworten Allen antwo... Weiterleiten Löschen Spam Markieren Mehr

Posteingang 8

Suchen ...

System Heute 18:45

- Bald ablaufende Produkte

System Heute 18:36

- Bald ablaufende Produkte

Katie | ClickUp Do 19:08

- ClickUp Partnership Programs

Katie | ClickUp 2021-11-22 19:06

- we need your advice

ClickUpdates 2.107 2021-11-22 18:40

- Whiteboards [Beta], Profile Cards, Click t...

Bald ablaufende Produkte

Von System am 2021-12-08 18:45

Details Einfacher Text

Liste der Produkte die in 4 Tage ablaufen:

- SW10008
- SW10003.1
- SW10000.3

Liste der Produkte die als TIPP im Shop angezeigt werden und in 14 Tage ablaufen:

- SW10000.2
- SW10003.2

## 9.12. Anzeigen der Mindesthaltbarkeit nach der Lieferung für frische Produkte


[Beschreibung](#) [Bewertungen](#)


### Produktinformationen "Mandeln-Schoko Torte"

**Mindesthaltbar nach der Lieferung:** Bis 7 Tage bei 0 Grad

Shoko-Mandeln runde Torte für 12 Personen

## 9.13. Anzeigen des Mindesthaltbarkeitsdatums für langhaltbare Produkte




**3,99 €\***  4,99-€\* (20.04% gespart)

Preise inkl. MwSt. zzgl. Versandkosten


- Mindesthaltbarkeitsdatum : 22. Dezember 2021
- Versandkostenfrei

Varianten

Variante 2(rot,rot-weiß)





In den Warenkorb


 [Zum Merkzettel hinzufügen](#)

Produktnummer: SW10003.2

## 9.14. Anzeigen von TIPP Produkte

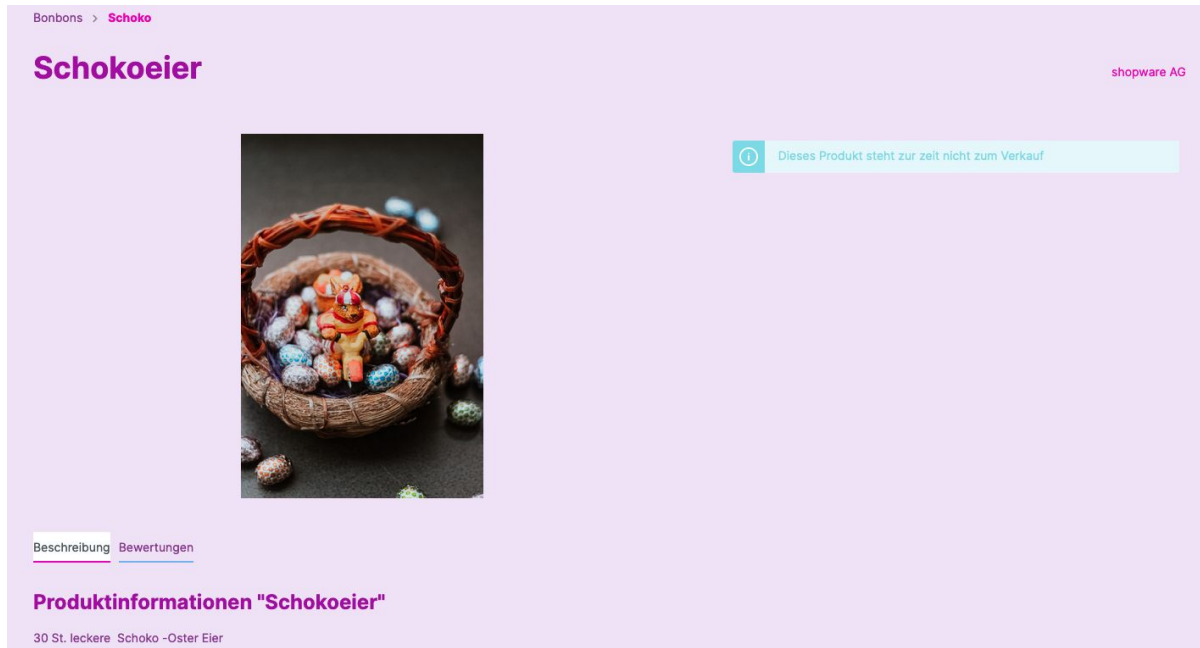
  
**Tipp**





Lolly

## 9.15. Anzeigen des Produkts im Shop bei ausgewählten Zusatzoption



## 9.16.Quellen:

<https://developer.shopware.com/docs/guides/plugins/plugins/plugin-fundamentals>

<https://academy.shopware.com/>

<https://gitter.im/shopware/platform?at=5e1eedf994656d7d5705407d>

<https://forum.shopware.com/>

<https://www.shopware.com/de/produkte/shopware-6/>

<https://www.8mylez.com>

<https://www.gehaltsvergleich.com/>