# Subscriptions - DRAFT

* [Overview](#Subscriptions-DRAFT-Overview)
* [Functional Requirements](#Subscriptions-DRAFT-FunctionalRequireme)
* [Technical Requirements](#Subscriptions-DRAFT-TechnicalRequiremen)
  + [Functionality that still needs Tech Design](#Subscriptions-DRAFT-Functionalitythatst)
    - [Pricing](#Subscriptions-DRAFT-Pricing)
  + [Sequence Diagrams](#Subscriptions-DRAFT-SequenceDiagrams)
    - [Add Subscription](#Subscriptions-DRAFT-AddSubscription)
    - [Update Subscription](#Subscriptions-DRAFT-UpdateSubscription)
    - [Placing a Subscription Order](#Subscriptions-DRAFT-PlacingaSubscriptio)
    - [Fulfill Subscription Order](#Subscriptions-DRAFT-FulfillSubscription)
    - [Cancel Subscription](#Subscriptions-DRAFT-CancelSubscription)
  + [Class Diagrams](#Subscriptions-DRAFT-ClassDiagrams)
    - [GPSubscription Services](#Subscriptions-DRAFT-GPSubscriptionServi)
    - [Subscription Cart/Order Classes](#Subscriptions-DRAFT-SubscriptionCart/Or)
    - [GPSubscriptionProduct](#Subscriptions-DRAFT-GPSubscriptionProdu)
* [Example Screenshots from Amazon](#Subscriptions-DRAFT-ExampleScreenshotsf)
  + [AWS Subscription PDP](#Subscriptions-DRAFT-AWSSubscriptionPDP)
  + [AWS Subscription Order Page](#Subscriptions-DRAFT-AWSSubscriptionOrde)
  + [AWS Subscriptions Order Confirmation Page](#Subscriptions-DRAFT-AWSSubscriptionsOrd)
  + [AWS Cancel Subscription Order Page](#Subscriptions-DRAFT-AWSCancelSubscripti)

This is currently a draft of Subscription implementation.  Once it is finalized the FDS/TDS will be generated in Word and uploaded to Sharepoint.

# Overview

This document describes the initial draft to support the MVP Subscription code with Hybris.  Amazon's flow was chosen as the example for UX, though we are not building all the functionality available on AWS.  Please see the screen captures at [Amazon Screen Captures](#Subscriptions-DRAFT-amazonexamples)

# Functional Requirements

|  |
| --- |
| **Requirement** |
| A guest and logged in user can select from a predefined list of subscription durations but must be logged in to subscribe. |
| Only one product/kit can be subscribed to at a time.  User will be able to select the quantity per subscription item. |
| A user must be logged-in to complete a subscription order.  If the user is not logged in, they will be redirected to the log in screen and on successfully log in redirected to the Subscription Order Confirmation screen.  **<Dev Comments> - We are going to redirect user to log in screen on click of Subscribe** |
| * Hybris will support variable frequencies for subscriptions for each product. * There will only be 1 discount/price per multiple frequencies per product.   **<Dev Comments> - Need to check for Discount rows as currently we have created one price row per frequency per product.**   * Frequency duration will be defined in weeks, but a free text display name will be available for the merchandiser to use.   **<Dev Comments> - Currently storing duration as Enumeration values e.g. DAILY, WEEKLY, MONTHLY and Subscription schedules are automatically getting calculated based on frequency Enumeration.**   * The system will only apply the number of weeks, it will not necessarily be the same day each frequency * E.G., Apply 5% off on the following frequencies   + Monthly (4 weeks)     - starts on Aug 20th 2018, the next deliver would be 4 weeks later September, 17th 2018   + Weekly ( 1 week) |
| Based on the selected subscription duration, hybris will display the price of the order. |
| A GP Business User can configure the subscription frequency options by product.  **<Dev Comments> - Business user will create different price rows for different frequencies per product.** |
| A GP business user can configure a % or $ amount off of this single order and/or future orders for a subscription.  **<Dev Comments> - User can apply appropriate coupon based promotion for a specific order in subscription.** |
| Hybris will maintain subscription pricing and promotions. |
| Hybris will create the order and send to S/4 for fulfillment n days before the subscription is due  **<Dev Comments> - A cart will be converted to Order based on subscription frequency.** |
| Hybris will send subscription notification email n days prior to sending order to S/4 / NetSuite  **<Dev Comments> - Not part of subscription addon.** |
| Order placed email will be sent when cronjob sends order to S/4, NetSuite   * This can be the same email as a normal order, with some slightly different content based on it being a subscription order   **<Dev Comments> - Not part of subscription addon.** |
| Order shipped email will be sent when fulfilled   * This can be the same email as a normal order, with some slightly different content based on it being a subscription order   **<Dev Comments> - Not part of subscription addon.** |
| A user will receive an email alert if an item in their replenishment order has been substituted or discontinued.  **<Dev Comments> - Not part of subscription addon.** |
| Registered users can receive SMS alerts for subscription notifications if SMS service is available in R2  **<Dev Comments> - Not part of subscription addon.** |
| A user will not be unsubscribed from a subscription order if a replacement product in their order has been substituted.  **<Dev Comments> - Substitution logic should take care of replacing the product in subscription cart.** |
| A logged-in user can only cancel a single order within a subscription  **<Dev Comments> - As of now handled through backoffice.** |
| Hybris will ensure that PCI/PII compliance is maintained due to how long we store customer information/tokening, etc. related to subscriptions.  **<Dev Comments> - Not part of subscription addon.** |
| A guest or logged-in user can filter and browse subscription products within search by indexing the isSubscribable attribute  **<Dev Comments> - Not part of subscription addon.** |

# Technical Requirements

## Functionality that still needs Tech Design

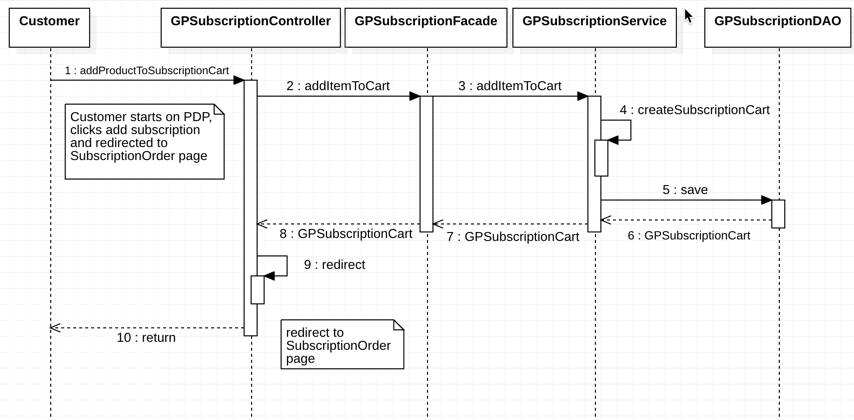
### Pricing

* The discount to be applied is associated with the product.  It will need to be applied to the cart total calculation before tax & promotions are calculated.

**<Dev Comments> - Only coupon based promotions are handled through subscription addon.**

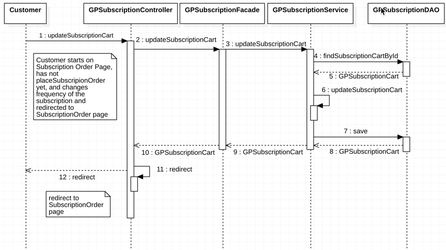
## Sequence Diagrams

### Add Subscription



1. Customer visits a PDP page of a subscrible product
   1. The page shows a dropdown with all the subscription options
   2. Customer selects 1 option and clicks the subscribe button
2. GPSubscriptionController validates inputs including
   1. Product code
   2. Frequency of Subscription
   3. Quantity
3. GPSubscriptionFacade
4. GPSubscriptionService creates a GPSubscriptionCart to hold the GPSubscriptionProduct
   1. GPSubscriptionCart and GPSubscriptionProduct are subclasses of their Hybris counterparts and therefore should be able to run through the normal cart processes like calculation
5. GPSubscriptionDAO persists the GPSubscriptionCart
6. returns GPSubscriptionCart
7. returns GPSubscriptionCart
8. GPSubscriptionFacade
9. GPSubscriptionController redirects to the Subscription Order Page

### Update Subscription



1. Customer is on the Subscription Order page, but has NOT placed the Order yet
   1. The page shows a dropdown with all the subscription options, but current frequency option is selected
   2. Customer changes the frequency option

**<Dev Comments> - Updating Subscription frequency API is not available in Addon.**

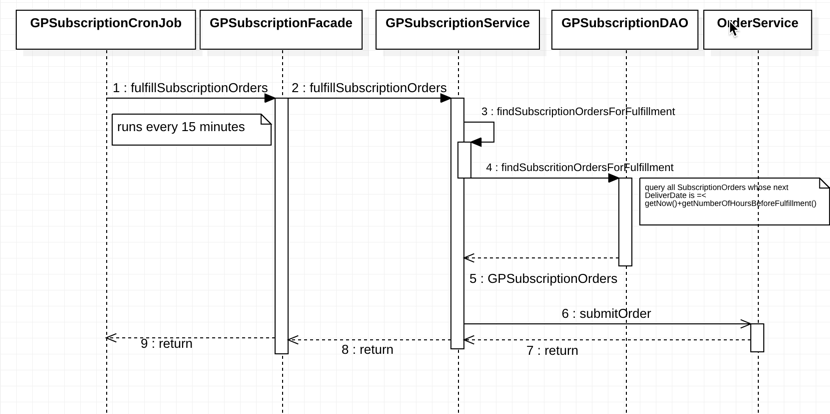
1. GPSubscriptionController validates inputs including
   1. Product code
   2. GPSubscriptionCart.code
   3. New Frequency of Subscription
   4. New Quantity
2. GPSubscriptionFacade
3. GPSubscriptionService calls GPSubscriptionDAO to find the subscription cart
4. GPSubscriptionCart is returned
5. GPSubscriptionService
   1. iterates through the cart to fine the Product code
   2. updates the Frequency, quantity
6. GPSubscriptionService the GPSubscriptionCart
7. returns GPSubscriptionCart
8. returns GPSubscriptionCart
9. returns GPSubscriptionCart
10. GPSubscriptionController redirects to the Subscription Order Page

### Placing a Subscription Order

## Place Subscription Order

1. Customer is on the Subscription Checkout Page and clicks Confirm Subscription
2. GPSubscriptionController validates inputs including
   1. GPSubscriptionCart.code
3. GPSubscriptionFacade
4. GPSubscriptionService creates a GPSubscriptionOrder
   1. Sets the cart attribute to GPSubscriptionCart
   2. Sets the nextDeliveryDate by using the GPFrequency.durationsInWeeks + getNow()
   3. GPSubscriptionCart and GPSubscriptionProduct are subclasses of their Hybris counterparts and therefore should be able to run through the normal cart processes like calculation
5. GPSubscriptionOrder is persisted
6. returns GPSubscriptionOrder
7. returns GPSubscriptionOrder
8. returns GPSubscriptionOrder
9. GPSubscriptionController redirects to the Subscription Order Confirmation Page
10. Subscription Order Confirmation Page is displayed

### Fulfill Subscription Order



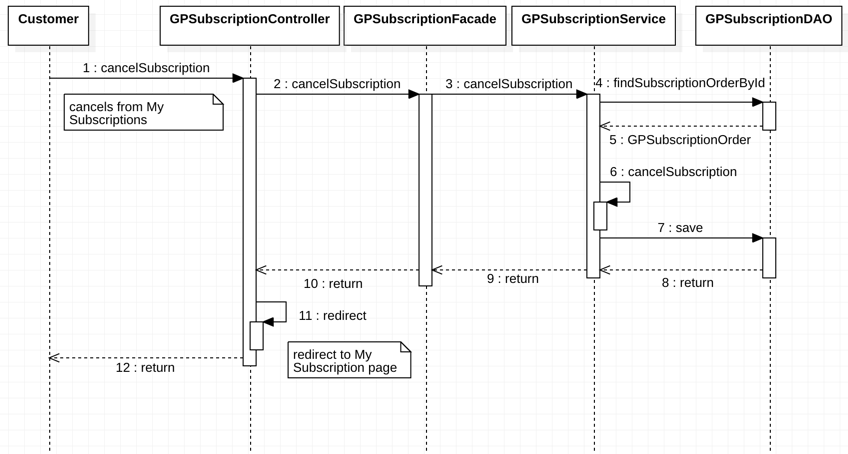
1. GPSubscriptionCronJob runs every 15 minutes
2. GPSubscriptionFacade
3. GPSubscriptionService calls GPSubscription DAO to get subscription orders ready to be fulfilled
4. GPDAO queries the database for
   1. query all SubscriptionOrders whose next DeliverDate is =< getNow()+getNumberOfHoursBeforeFulfillment()
      1. getNumberOfHoursBeforeFulfillment is the number of hours Hybris should send this subcription order to fulfillment
         1. E.G.,
            1. Subscription is weekly
            2. numberOfHoursBeforeFulfillment is 24 hours
            3. System should send the subscription order 1 day before the nextDeliveryDate

**<Dev Comments> - Order fulfillment is not handled in Subscription addon.**

1. GPSubscriptionService receives a Collection of GPSubscriptionOrders for fulfillment and iterates over each GPSubscriptionOrder
   1. Converts the GPSubscriptionCart into a valid Order object
      1. we need to set flag isSubscription for later reporting
   2. Try/Catch should be inside the loop, so a order that fails won't prevent any other order from being processed
2. Sends order Order Service to be submited to the Order workflow
3. Returns
4. Returns

**\*Note:  Step 6 is the similiar as if the customer when through normal checkout and clicked Place Order.  All emails and notifications should follow as per a regular order.  The exception is that there might be small content changes based on the fact this Order was generated from a subscription**

### Cancel Subscription

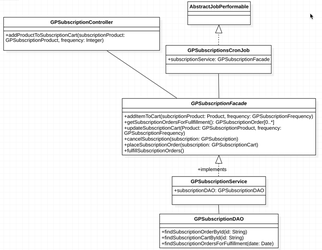


1. Customer visits the My Subscriptions in My Account
   1. The page shows all the subscription orders
   2. Customer selects clicks the cancel button button
2. GPSubscriptionController validates inputs including
   1. SubscriptionOrder code
3. GPSubscriptionFacade
4. GPSubscriptionService calls GPSubscriptionDAO to find the subscription cart
5. GPSubscriptionDAO returns GPSubscriptionOrder
6. GPSubscriptionService cancels GPSubscriptionOrder by
   1. Setting state to CANCELLED
   2. Setting the nextDeliveryDate to null
7. GPSubscriptionOrder is persisted
8. returns
9. returns
10. returns
11. GPSubscriptionController redirects to My Subscription Page
12. My Subscription Page now showing the GPSubscriptionOrder as cancelled

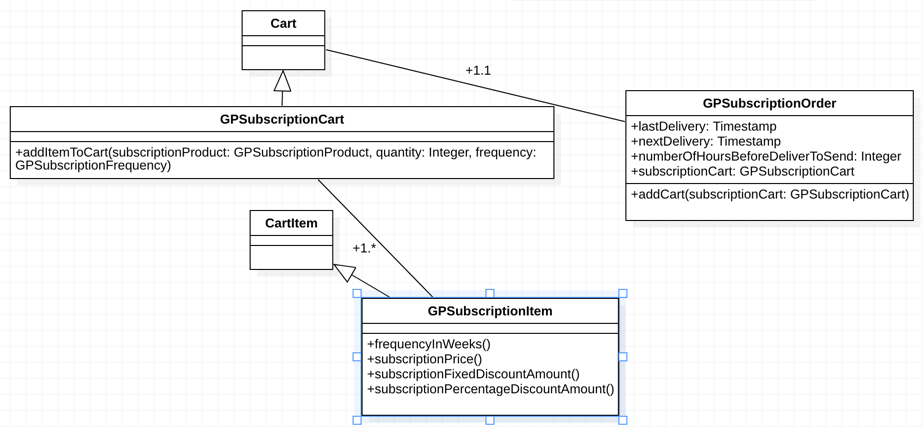
**<Dev Comments> - Subscription cancellation is handled through Backoffice.**

## Class Diagrams

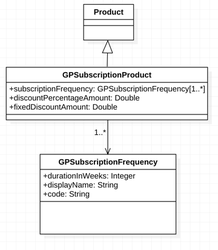
### GPSubscription Services



### Subscription Cart/Order Classes



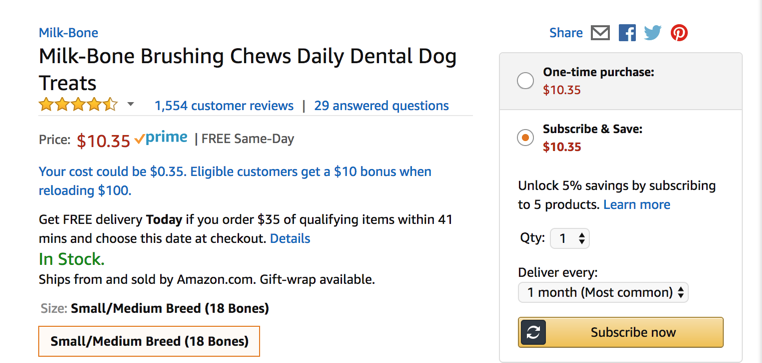
### GPSubscriptionProduct



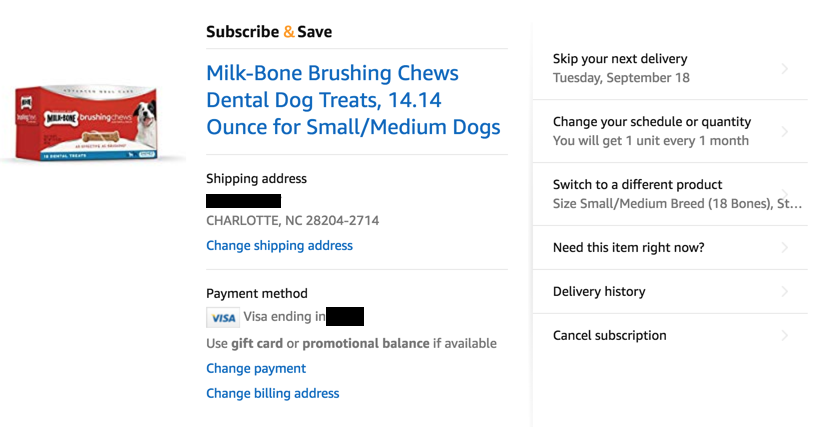
# Example Screenshots from Amazon

The screenshots below are for discussion only.  The screens below are showing more functionality than GP will implement.

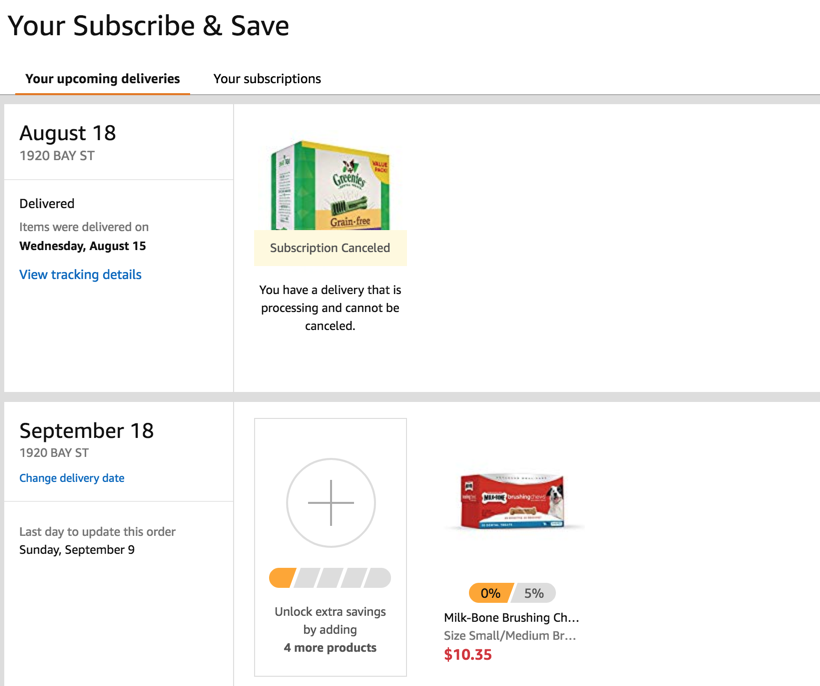
## AWS Subscription PDP



## AWS Subscription Order Page



## AWS Subscriptions Order Confirmation Page



## AWS Cancel Subscription Order Page

