**Stripe Cartridge**

**Integration guide**

[1. Controllers 4](#_Toc23241150)

[1.1 Summary 4](#_Toc23241151)

[1.2 Component Overview 4](#_Toc23241152)

[1.3 Setup 4](#_Toc23241153)

[1.3.1 Configuration 4](#_Toc23241154)

[1.4 ApplePay Settings 10](#_Toc23241155)

[1.5 Scheduled Job 12](#_Toc23241156)

[1.6 Code modifications 12](#_Toc23241157)

[cartridge\scripts\checkout\ValidatePaymentInstruments.ds 12](#_Toc23241158)

[cartridge\templates\default\account\payment\paymentinstrumentdetails.isml 13](#_Toc23241159)

[cartridge\templates\default\account\payment\paymentinstrumentlist.isml 13](#_Toc23241160)

[cartridge\templates\default\checkout\billing\paymentmethods.isml 13](#_Toc23241161)

[cartridge\templates\default\checkout\cart\cart.isml 14](#_Toc23241162)

[cartridge\templates\default\checkout\confirmation\confirmation.isml 14](#_Toc23241163)

[cartridge\templates\default\components\footer\footer\_UI.isml 14](#_Toc23241164)

[cartridge\templates\default\components\order\orderdetails.isml 14](#_Toc23241165)

[cartridge\templates\default\components\order\orderdetailsemail.isml 15](#_Toc23241166)

[cartridge\templates\default\components\order\ordertotals.isml 15](#_Toc23241167)

[cartridge\controllers\COBilling.js 15](#_Toc23241168)

[cartridge\controllers\COPlaceOrder.js 16](#_Toc23241169)

[cartridge\controllers\PaymentInstruments.js 17](#_Toc23241170)

[2. SFRA 18](#_Toc23241171)

[2.1 Implementation Guide 18](#_Toc23241172)

[2.1.1 Compile static resources 18](#_Toc23241173)

[2.1.2 Assign cartridges 18](#_Toc23241174)

[2.2 Code modifications 19](#_Toc23241175)

[cartridge\controllers\Account.js 19](#_Toc23241176)

[cartridge\controllers\CheckoutServices.js 19](#_Toc23241177)

[cartridge\controllers\PaymentInstruments.js 21](#_Toc23241178)

[cartridge\controllers\RedirectURL.js 21](#_Toc23241179)

# Controllers

# Summary

This document describes the integration process of Stripe with Demandware (SiteGenesis) application.

In order to use the Stripe Cartridge, customers must have an account in Stripe, testing mode is available.

The integration will require the customer to deploy the provided cartridge and update a number of controllers and templates to incorporate the functionality provided by the Stripe Cartridge.

# Component Overview

The stripe sales force commerce cloud cartridge provides the integration of the following payment methods:

* Credit Card
* iDeal
* SEPA Direct Debit
* Sofort
* Alipay
* Bancontact
* EPS
* ACH Credit Transfer
* Giropay
* Przelewy24

# Setup

## Configuration

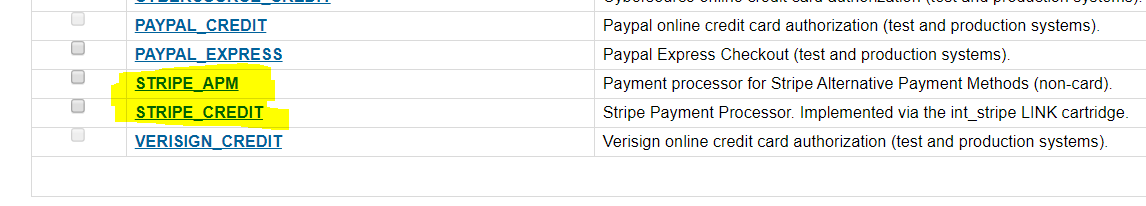
Before the Stripe functionality can become available to Reference Architecture, the cartridges have to be added to the cartridge path of the Site in question. In order to do this, follow the following instructions:

1. Log into Business Manager
2. Navigate to *Administration 🡪 Sites 🡪 Manage Sites*.
3. Click on the site name and on the next page go to the *Settings* tab.
4. In the textbox *Cartridges* add “**int\_stripe\_controllers:int\_stripe\_core:sitegenesis\_storefront\_controllers:sitegenesis\_storefront\_core**”.
5. Click *Apply*.
6. To activate the cartridge for the Sandbox instances repeat steps **4** and **5** after selecting the appropriate instance from the *Instance Type* dropdown menu.

Repeat steps **3** to **6** for each site that is to use Stripe.

## Import metadata

Import the metadata from the metadata folder.

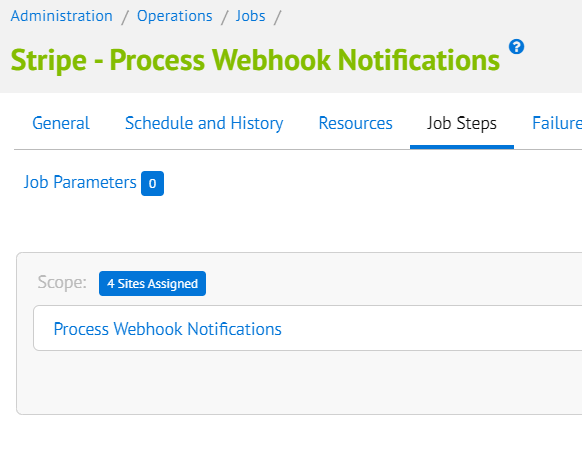
Two new payment processors – ‘STRIPE\_CREDIT’ and ‘STRIPE\_APM’. 

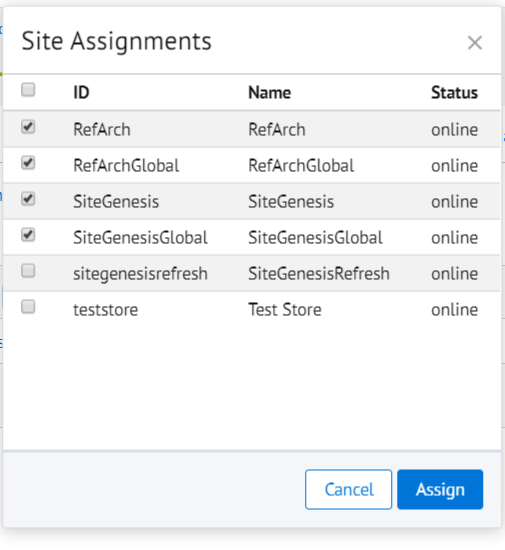
A new service, “**stripe.http.service**” has been created as part of the integration.

There are 2 jobs coming with the package:

* Stripe - Delete Custom Objects
* Stripe - Process Webhook Notifications

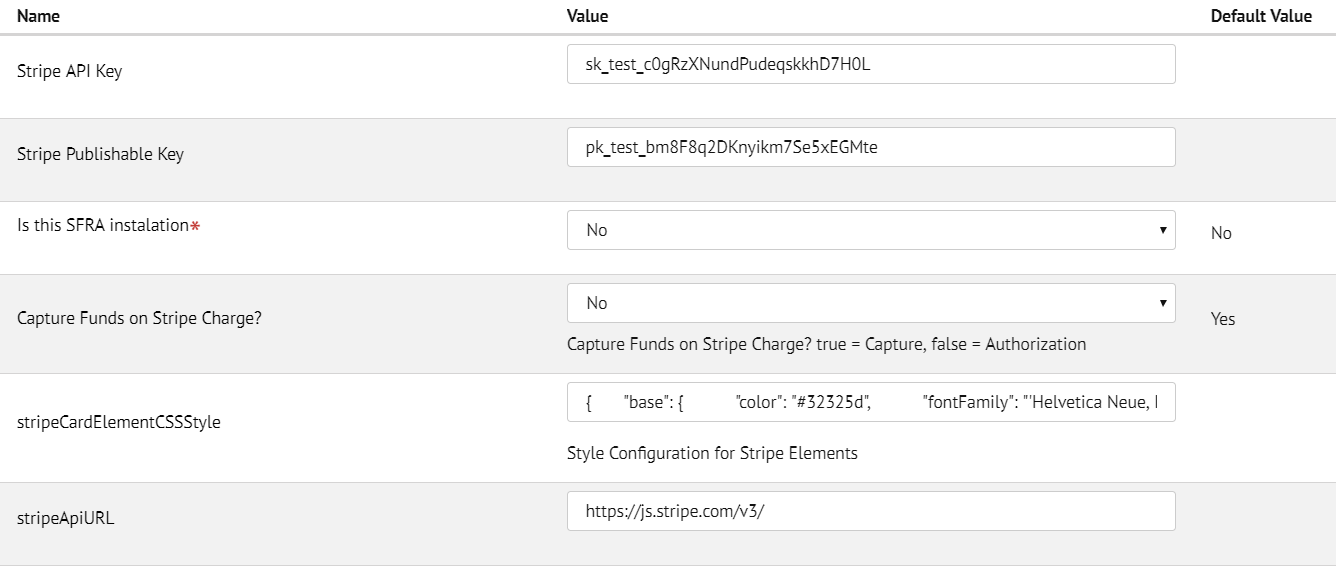
Enable the job “Stripe - Process Webhook Notifications” for the desired site:

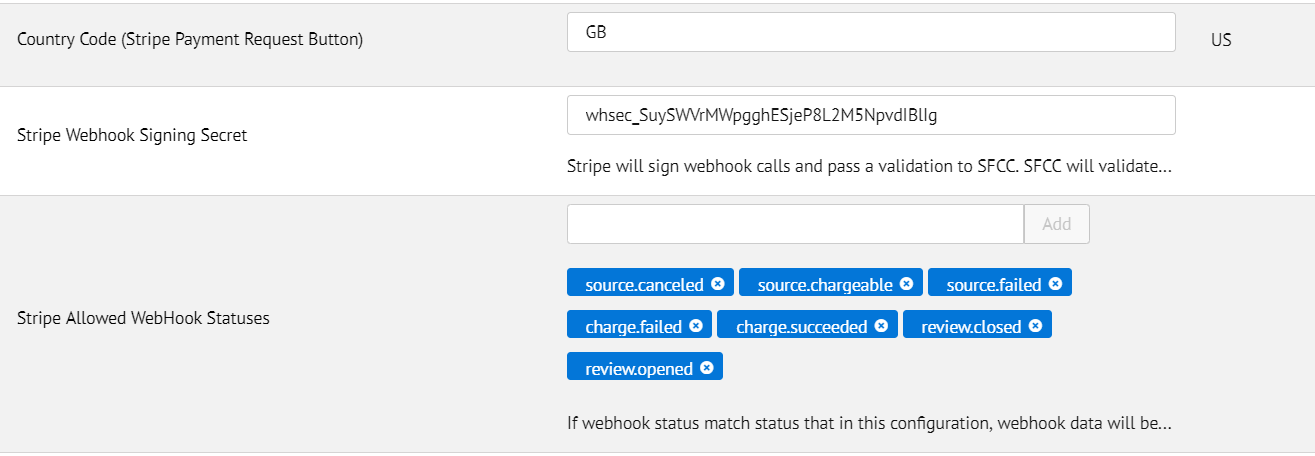


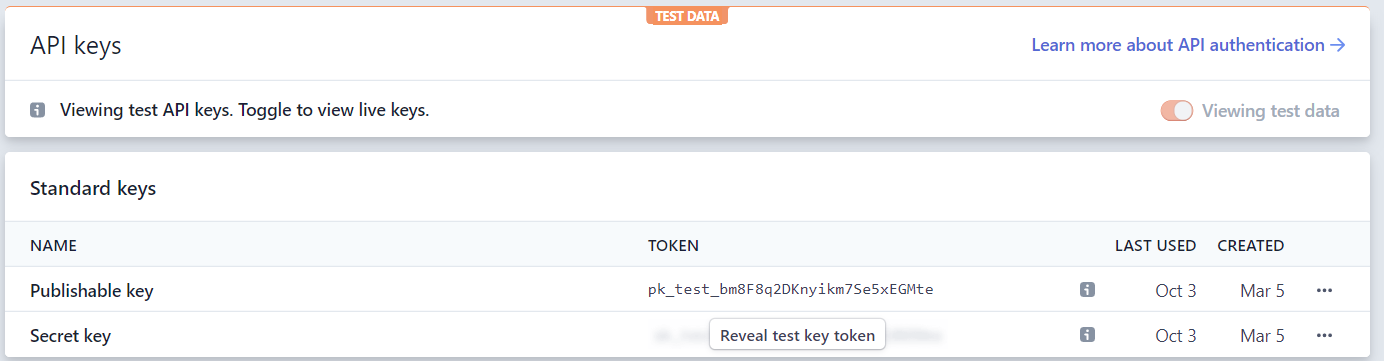


In Business Manager, navigate to the *MoblieFirst Site🡪* Merchant Tools🡪 *Site Preferences* 🡪 *Custom Preferences*. A custom site preference group with the ID **Stripe Configs** is available. Please select it and edit the attributes according to your Stripe account data:

<https://dashboard.stripe.com/test/apikeys>:

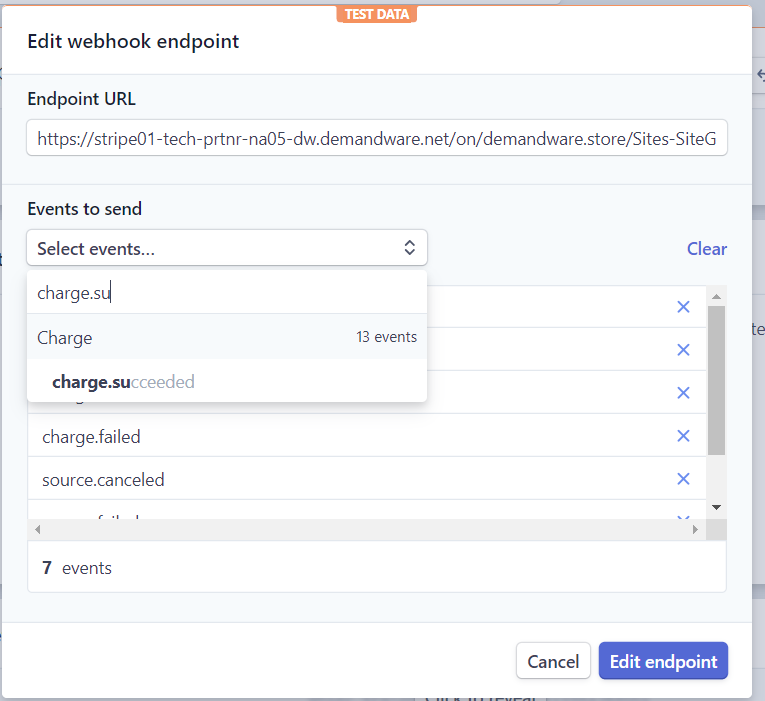


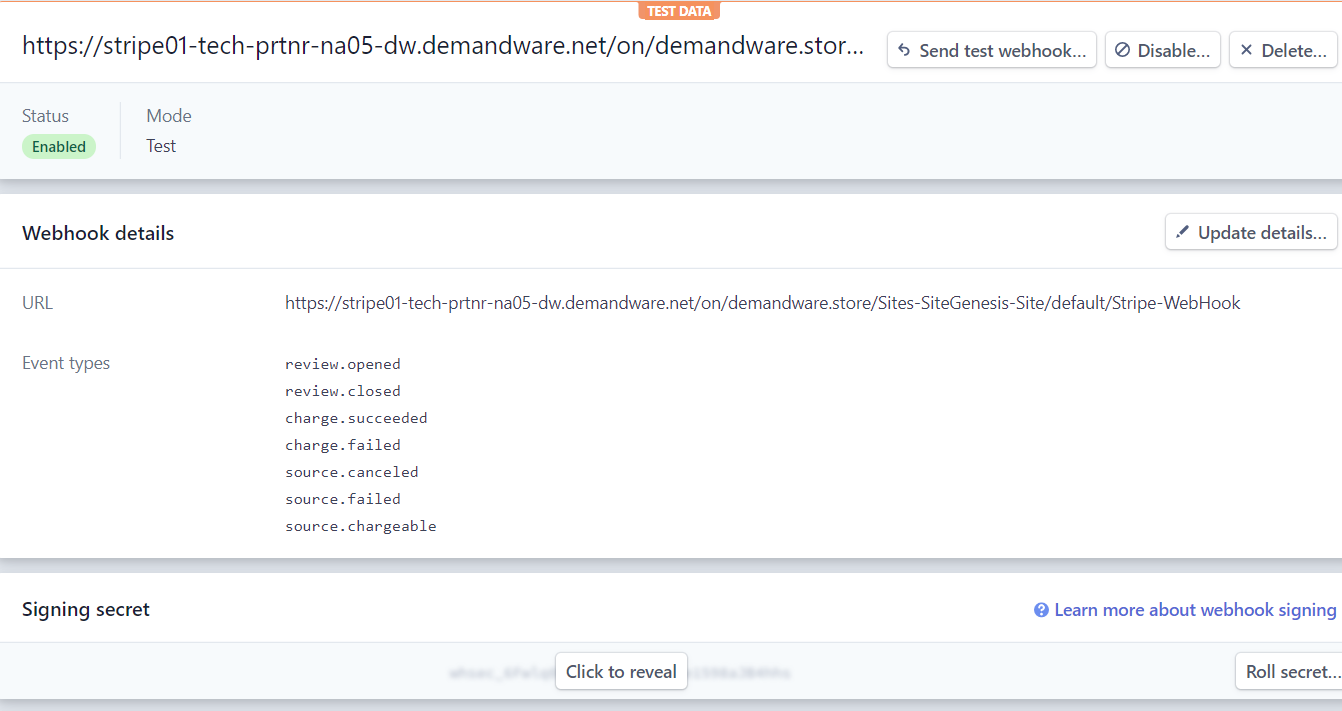




In Stripe Dashboard (<https://dashboard.stripe.com/test/webhooks>) enable webhooks, point it to `Stripe-WebHook` controller and subscribe to these events:

* review.opened
* review.closed
* charge.succeeded
* charge.failed
* source.canceled
* source.failed
* source.chargeable



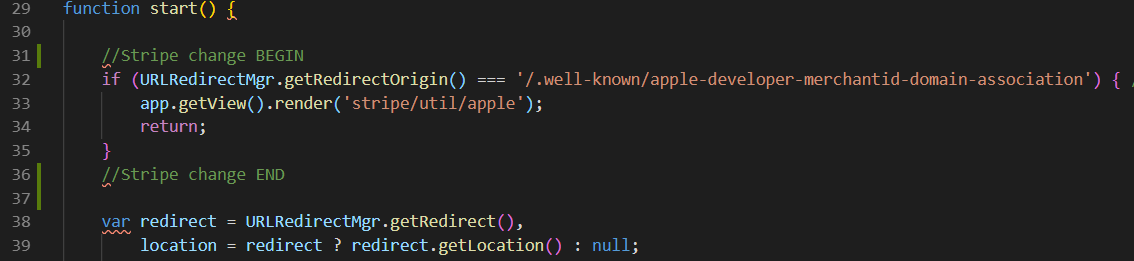


Then copy the signing secret to the ‘Stripe Webhook Signing Secret’ preference.

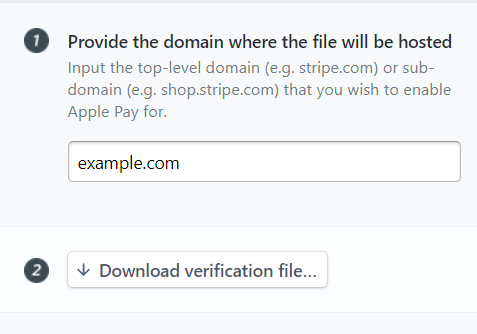
Make sure that this value is set to your Stripe account country code: 

# ApplePay Settings

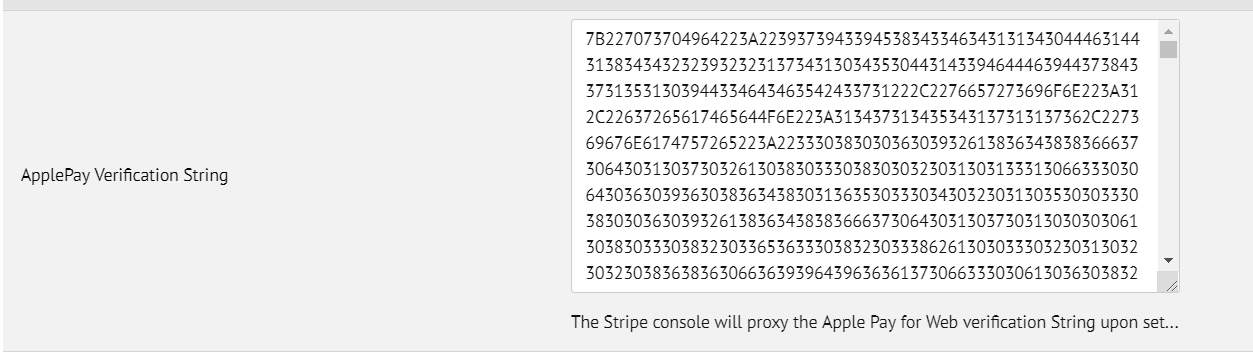
For ApplePay to work, the file RedirectURL.js must be changed with this code:



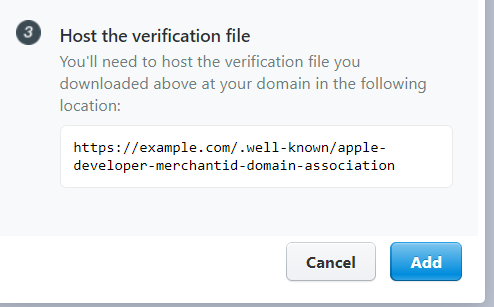
Then go to <https://dashboard.stripe.com/account/apple_pay> and click on ‘Add new domain’ button. Enter the domain and download the verification file:



Copy the contents of the file to ‘ApplePay Verification String’ custom preference:



Then click on the ‘Add’ button:



# Scheduled Job

The Scheduled job included in the cartridge handle the webhook notifications sent by Stripe. It will check if a source has become chargeable, charge it, verify that charge is successful and will place the order. If card payment is set in review by Radar rule and the payment is refunded, the job will fail the order. If the payment is confirmed, the order will be placed.

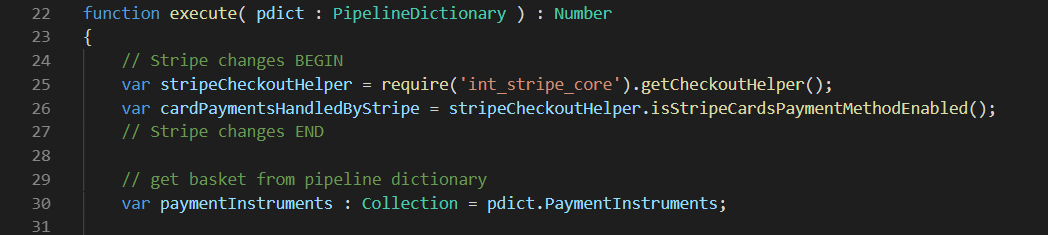
# Code modifications

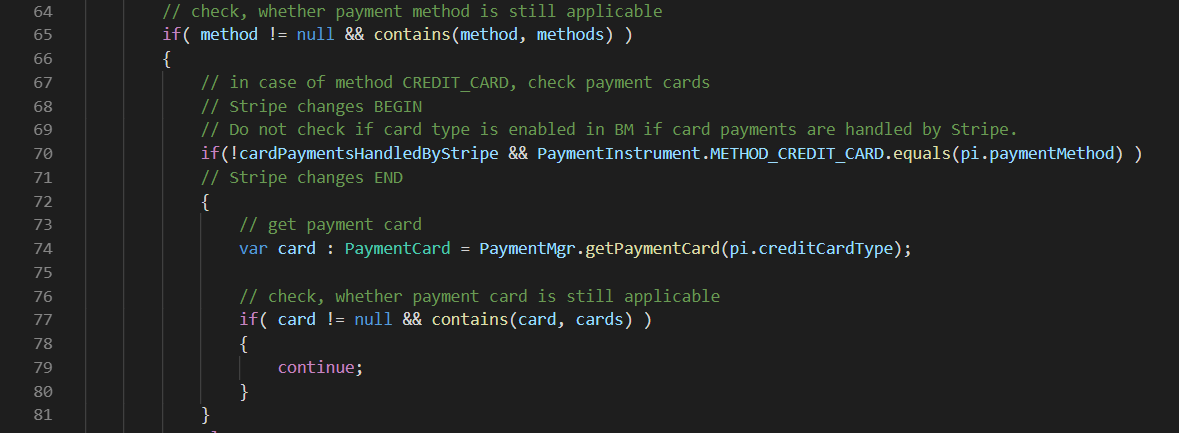
All code modifications are between:

// Stripe changes BEGIN and // Stripe changes END comments.

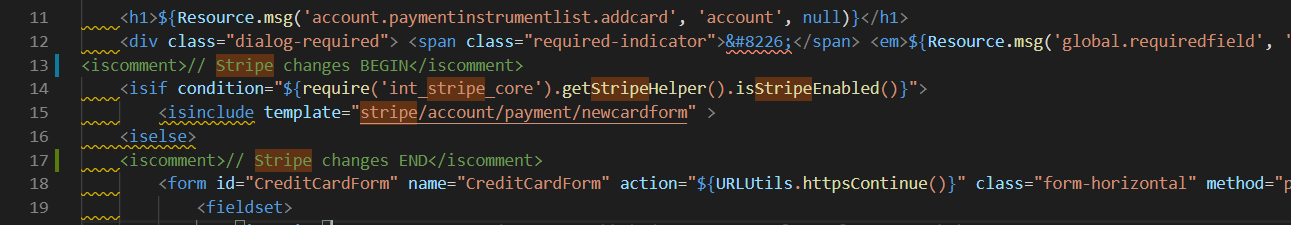
All files that need modifications are also included in ‘**app\_stripe\_core**` and `**app\_stripe\_controllers**` cartridges

### cartridge\scripts\checkout\ValidatePaymentInstruments.ds

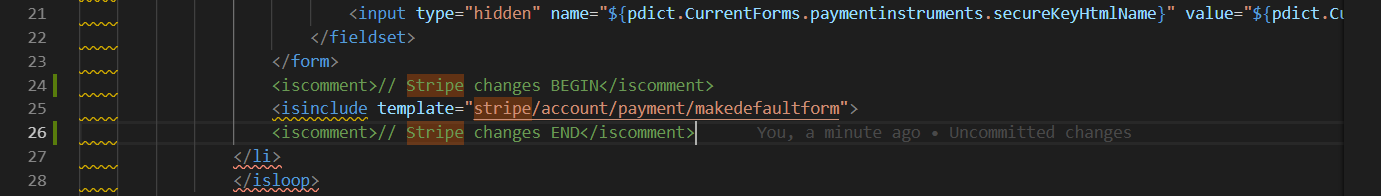




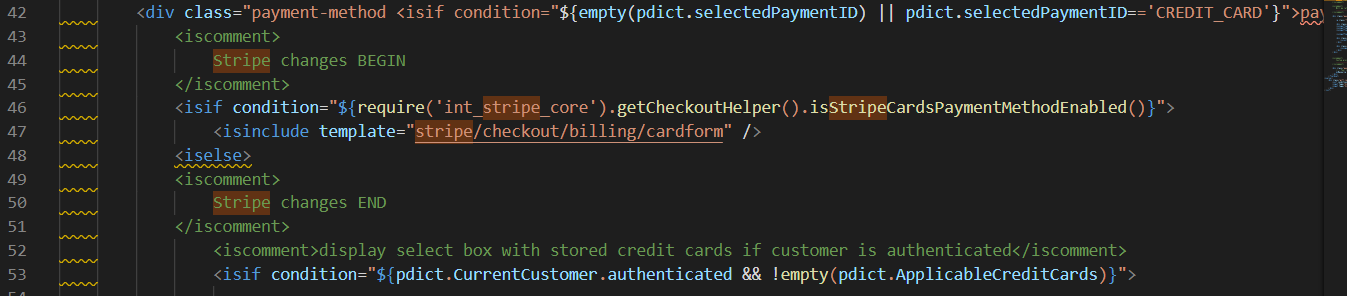
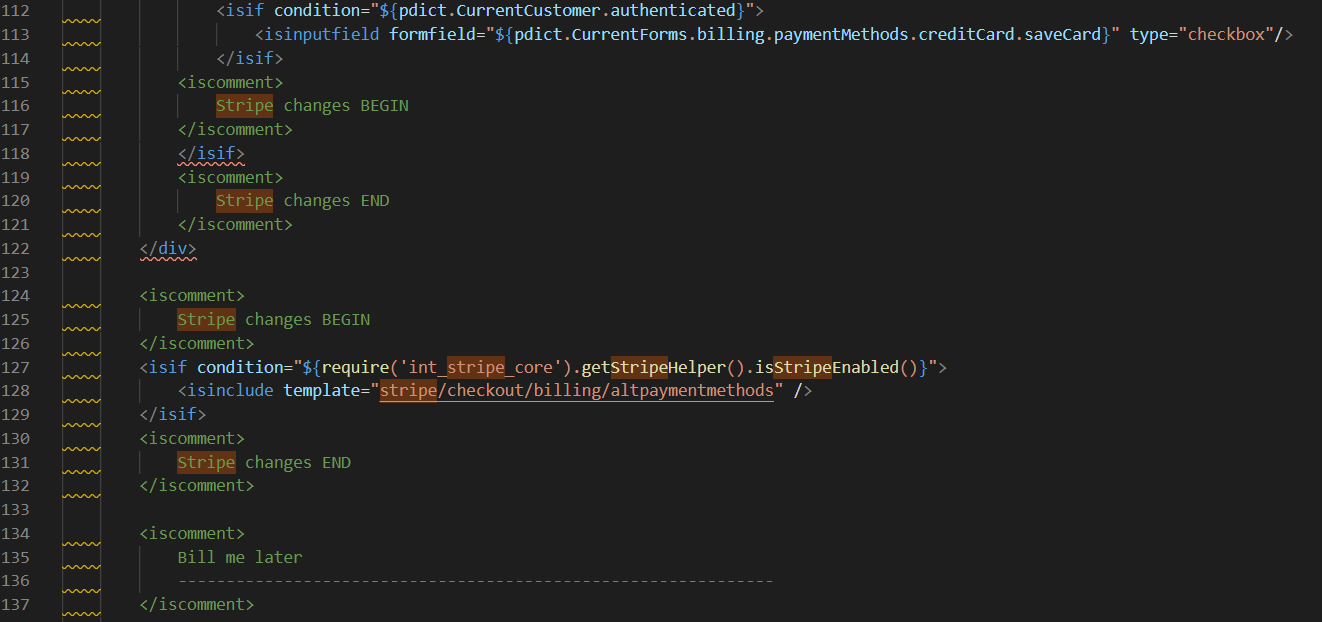
### cartridge\templates\default\account\payment\paymentinstrumentdetails.isml



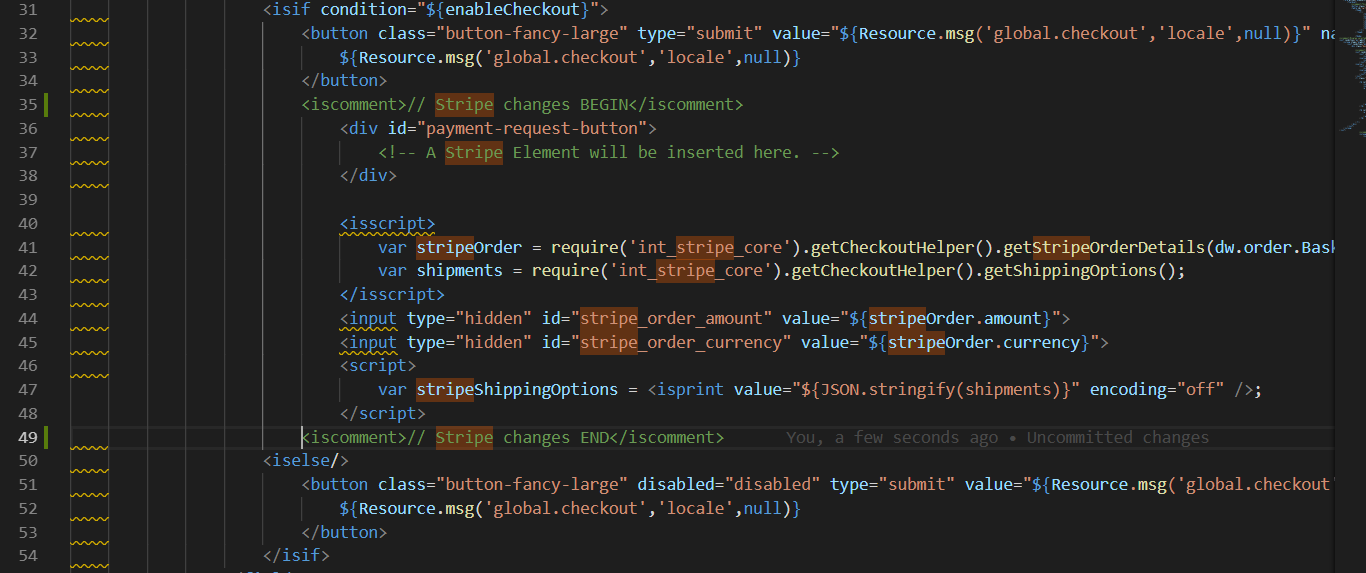
### cartridge\templates\default\account\payment\paymentinstrumentlist.isml



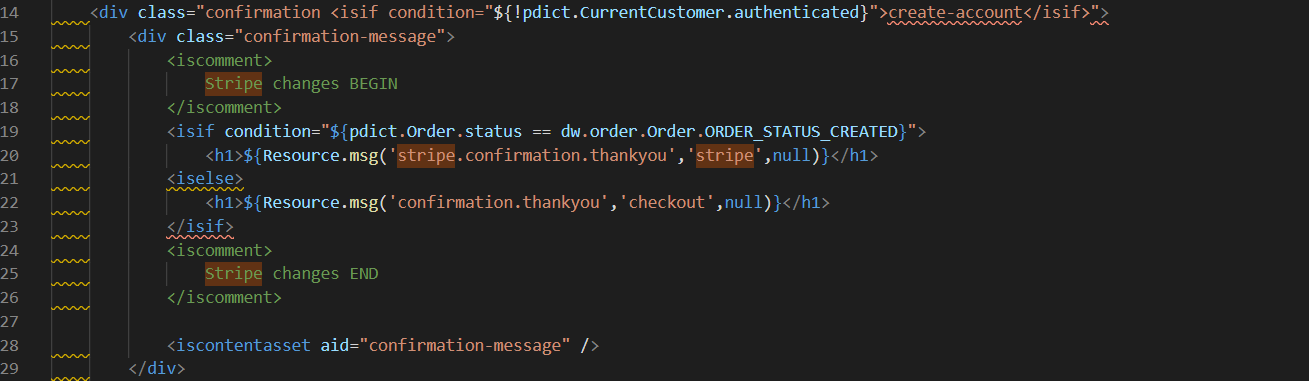
### cartridge\templates\default\checkout\billing\paymentmethods.isml

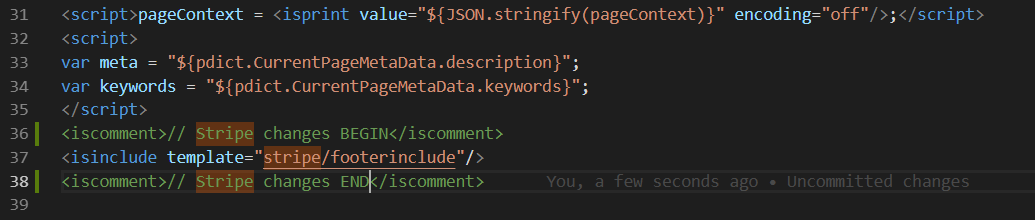
### cartridge\templates\default\checkout\cart\cart.isml



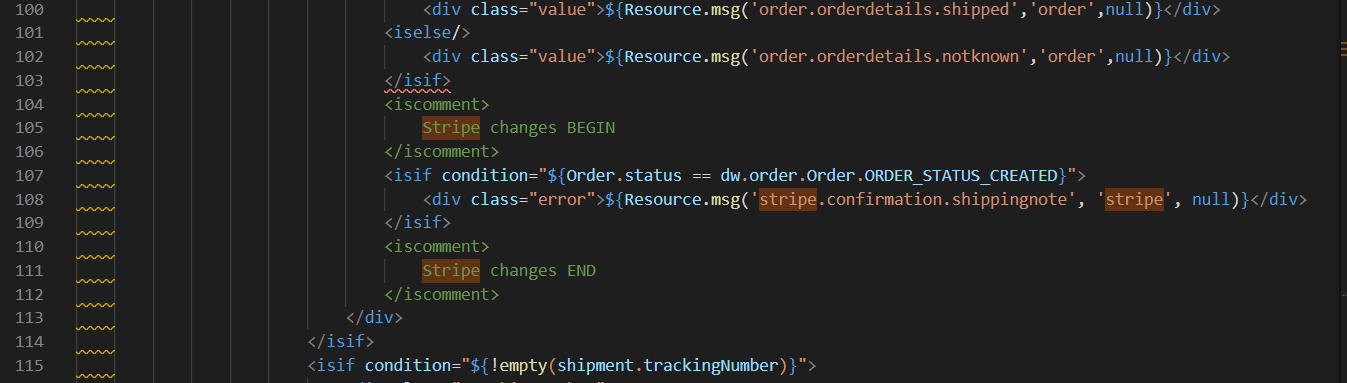
### cartridge\templates\default\checkout\confirmation\confirmation.isml



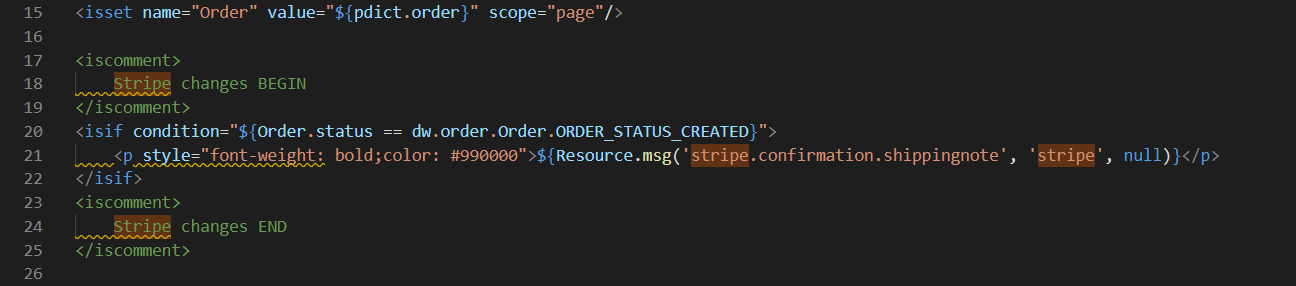
### cartridge\templates\default\components\footer\footer\_UI.isml



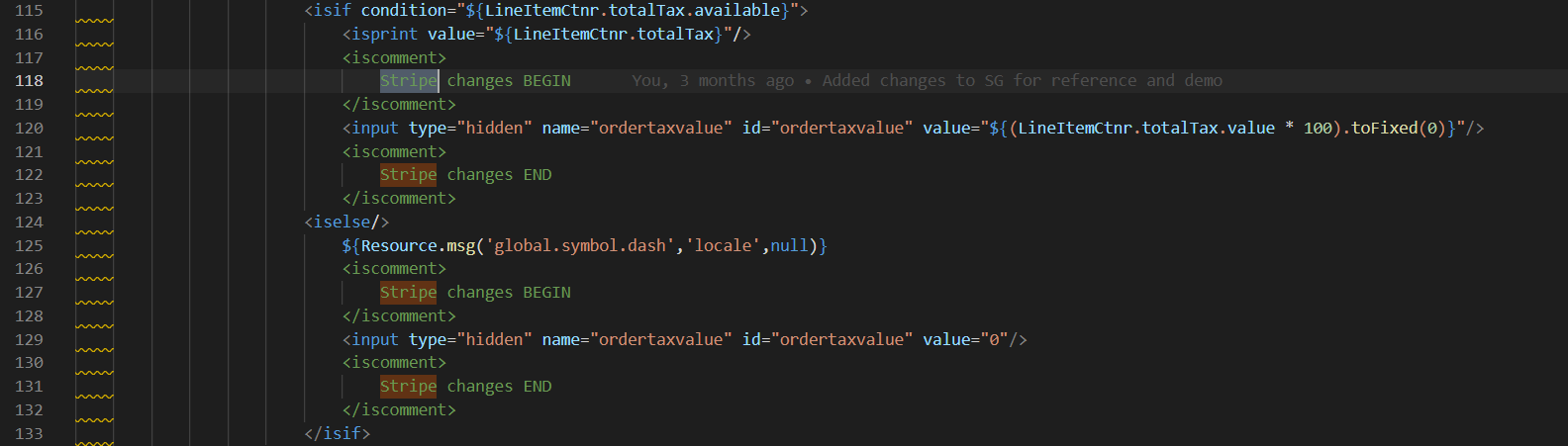
### cartridge\templates\default\components\order\orderdetails.isml



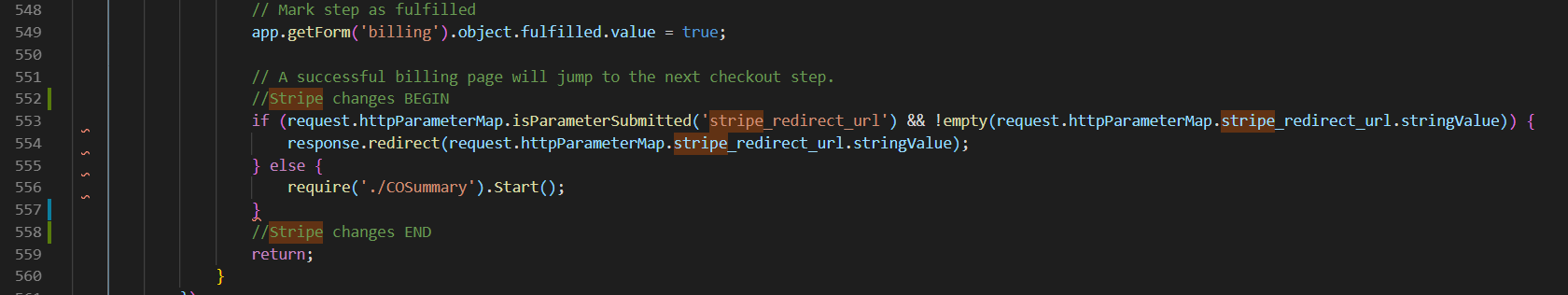
### cartridge\templates\default\components\order\orderdetailsemail.isml



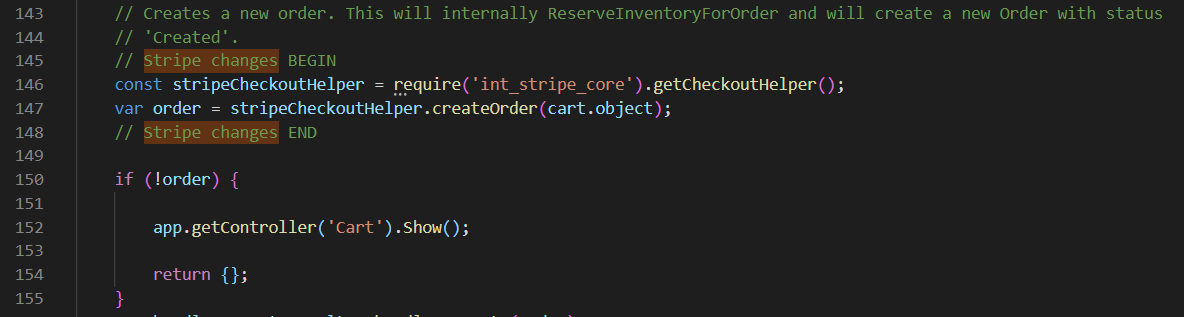
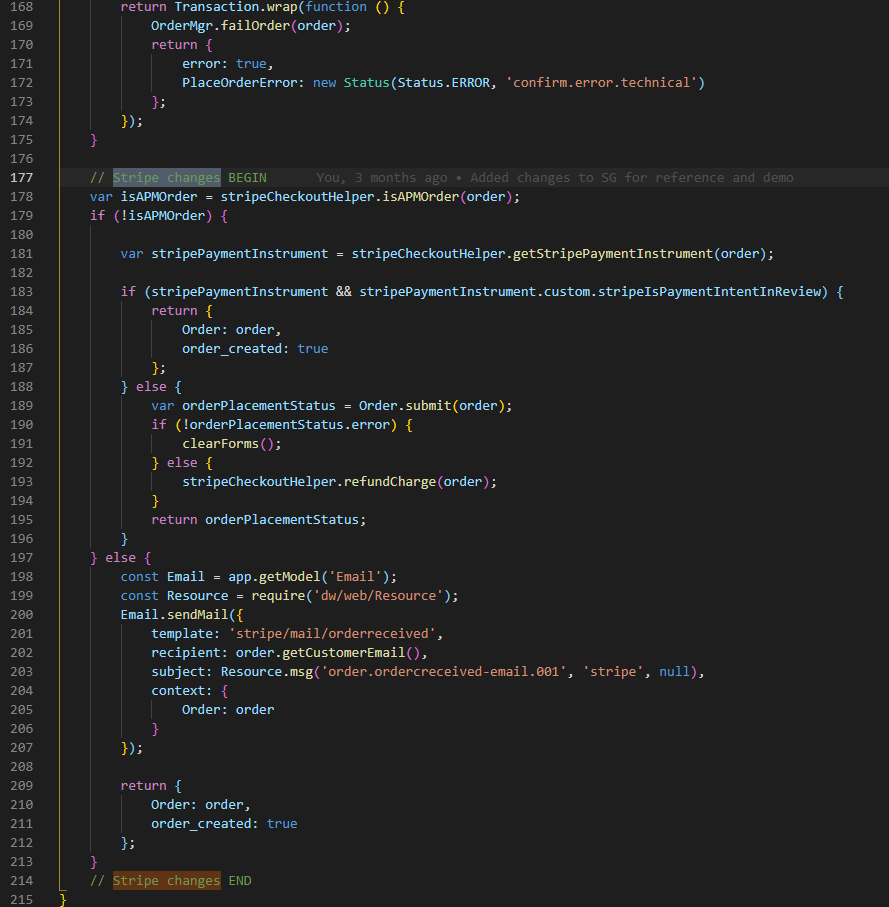
### cartridge\templates\default\components\order\ordertotals.isml



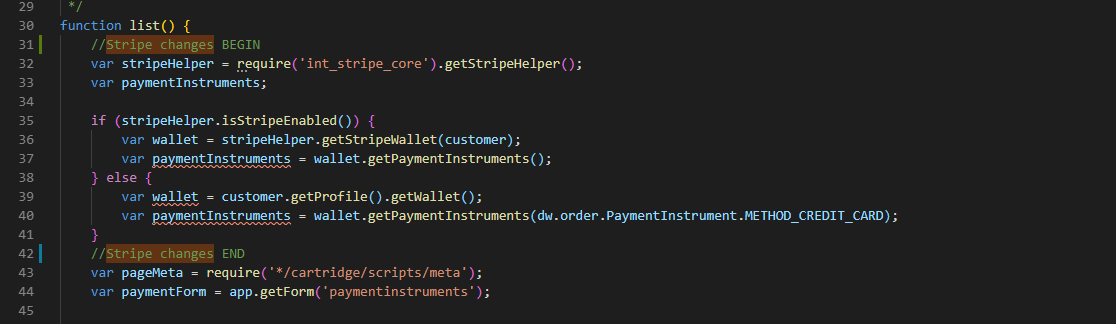
### cartridge\controllers\COBilling.js



### cartridge\controllers\COPlaceOrder.js

### cartridge\controllers\PaymentInstruments.js



# Operations, Maintenance

## Availability

Stripe services are expect to be available at all times. If a payment processor does not work, should there be a service failure, and the retailer will be notify via error email. Service connectivity errors will not be visible on the storefront, and will not affect the user’s ability to place orders. This is one of several payment processors and should be place orders with others.

# SFRA

# Implementation Guide

Stripe is compatible with SFRA 4.3 and greater.

# Compile static resources

From within the package‘s main directory run the following commands from the cli (command line):

• npm i

• npm run compile:js

• npm run compile:scss

# Assign cartridges

In the business manager follow these steps:

Navigate to Administration > Sites > Manage Sites

Choose the tab “Settings” of the site where you wish to add Wirecard Payment Gateway cartridges.

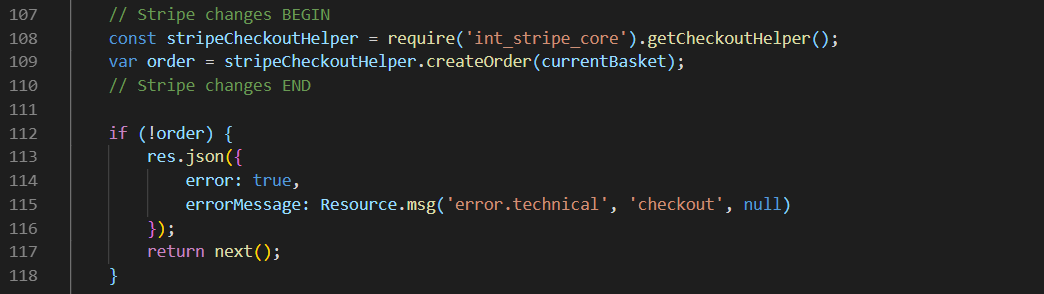
Append**: int\_stripe\_sfra:int\_stripe\_core:app\_storefront\_base** to the cartridge path.

# Code modifications

### cartridge\controllers\Account.js



### cartridge\controllers\CheckoutServices.js



### cartridge\controllers\PaymentInstruments.js



### cartridge\controllers\RedirectURL.js

