A close up of a logo

Description automatically generated

Version 23.3.0 for Salesforce Commerce Cloud - SFRA

Upgrade Guide

# 

# 

# Summary

This guide describes how to upgrade your existing Afterpay SFRA cartridge to v22.0.0 and above.

Afterpay’s Version 23.3.0 cartridge provides the following features:

* Provides the Afterpay or Clearpay payment method during checkout
* Provides the optional Afterpay or Clearpay Express Checkout feature
* Supports Afterpay’s v2.0 API
* Supports All Afterpay and Clearpay countries
* Provides Cash App Payment method for US Merchants
* Supports Cross-border trade for eligible merchants

# Table of Contents

[Summary](#_tyjcwt)

[Table of Contents](#_3dy6vkm)

[Component Overview](#_4d34og8)

[Upgrade Guide (for v20.0.0 and v21.0.0 cartridge users)](#_60cx1rb917o)

[Cash App Pay](#_qew35h81hvos)

[Testing](#_23ckvvd)

[Features and Usage](#_41mghml)

[Operations / Maintenance](#_4cyobfr9zh5y)

[Release History](#_3hv69ve)

# 

# Component Overview

## Functional Overview

Afterpay provides a payment processing gateway. Afterpay’s payment gateway connects your storefront to process transactions that allow customers to make purchases.

## Use Cases

Allows checkout using the Afterpay or Clearpay payment method at the checkout screen.

## Allows checkout using the Afterpay Express Checkout feature at the cart and product details screen.

## Allows checkout using Cash App payment methods for US merchants at the checkout screen in addition to the Afterpay Method.

## Limitations

The following Afterpay APIs are supported.

1. Direct Capture Payment
2. Authorize Payment
3. Create Checkout
4. Get Checkout
5. Get Configuration

The following Afterpay APIs are not supported or applicable

1. Void Authorization - Merchants will need to implement this in the event that orders can be cancelled from SFCC.
2. Capture Payment - If using our deferred payment flow the capture payment call will typically happen from an Order Management System
3. Update Shipping Courier - this will typically be implemented in the Order Management System.

The details about APIs for the different regions can be found here :

| Brand | Countries | URL |
| --- | --- | --- |
| **Afterpay** | **AU,US,CA,NZ** | <https://developers.afterpay.com/afterpay-online/reference> |
| **Clearpay** | **UK** | <https://developers.clearpay.co.uk/clearpay-online/reference> |
| **ES,IT,FR** | <https://developers.clearpay.com/v2/reference> |

## Compatibility

Compatibility mode version this was tested against is SFRA 5.3.0 & SFRA 6.3.0

## Privacy, Payment

This integration requires access to the following customer data elements:

* Billing Address
* Shipping Address
* Order Details
* Customer Profile.

No credit card details are stored within SFCC using this integration.

# 

# 

## Express Checkout

Express Checkout is a new feature which will reduce the overall checkout steps on desktop and mobile, so that your shoppers can complete their orders quickly. It enables shoppers to checkout directly from the shopping cart or product page; and use their previously provided information to Afterpay - name, shipping address, phone number, and email, to complete their orders on your website.

After adding Afterpay’s v23.1.0 cartridge, the following will be added to your site:

* “Checkout with Afterpay” buttons on the cart page
* “Buy Now with Afterpay” button on the product detail page .  
  (Optional but Highly Recommended)

When the shopper clicks on the checkout buttons, the following will occur:

* Express Checkout begins and Afterpay’s popup form appears.
* Shoppers who have used Afterpay in the past and have enabled saved shipping and payment information will be defaulted with those options and will not have to reenter that information, but can change as desired.
* Price and shipping methods will be presented to the shopper (with integrated shipping option enabled).
* Upon clicking the confirmation button, the shopper will be redirected back to the merchant site for review or immediate order creation (BuyNow option enabled)

## Cash App Pay

Cash App Pay Payment method option is provided for the US merchants. It enables Cash App consumers to checkout on Afterpay merchants’ website. When enabled Cash App Pay is shown at the checkout in addition to the Afterpay payment method for the eligible orders.

# Upgrade Guide (for v20.0.0 and v21.0.0 cartridge users)

## Download the v23.1.0 cartridge

The current version of the Afterpay cartridge is available directly from Afterpay.

<https://github.com/afterpay/afterpay-salesforce-commerce-cloud>

The version hosted on the Salesforce Commerce Cloud Partner Marketplace is an older version and should not be used.

## Build Client-Side Resources

This is required in order to build the client-side css and js files that are specific to Afterpay.

* Make sure package.json contains the correct reference to the location of the SFRA base cartridge:

| "paths": {  "base": "../storefront-reference-architecture/cartridges/app\_storefront\_base/" }, |
| --- |

* Run the following commands:

| $ npm install $ npm run compile:js $ npm run compile:scss |
| --- |

## Import the cartridge

Import the cartridge using UX Studio. Alternatively, can also use npm run uploadCartridge, or VSCode Prophet plugin.

## Import the site metadata

For the Afterpay integration to work, the following object structures (metadata) needs to be imported and configured in the Business manager. Follow the below steps:

1. In the cartridge bundle find metadata/afterpay-meta-import folder.
2. Find the sites folder and inside it find the RefArch folder.
3. Rename this RefArch folder to the ID of your site. If you have multiple sites to which this meta has to be imported, then copy and paste this RefArch folder to replicate for other sites. Renames those folders with ID of corresponding sites.
4. In the cartridge bundle, inside the metadata folder compress the afterpay-meta-import folder to generate the afterpay-meta-import.zip file.
5. Go to Business Manager Menu > Administration > Site Development > Site Import & Export
6. Under Import: Upload Archive : Ensure that the radio button with label Local is enabled (Else click on the radio button to enable it)
7. Click on Choose File input field, select the afterpay-meta-import.zip file from open dialog box and click on upload button
8. After finishing the upload, from the Archives list click the radio button corresponding to afterpay-meta-import.zip and click on Import button
9. Click on the OK button of the confirmation box asking “Are you sure that you want to import the selected archive?”

## Check For File Conflicts

Make sure the template, controller, and client JS and SCSS changes required for the cartridge do not interfere with changes you’ve made for your own site, or plugins that are running on your site.

The following templates are present in the Afterpay’s SFRA Cartridge as well as the app\_storefront\_base cartridge. Thus, the template in Afterpay’s cartridge will override the base template. Since Afterpay’s cartridge was built to work with the base cartridge, this should be fine. However, if you’ve made custom changes to these templates in the base cartridge, or have installed plugins which override these templates, you will need to merge the changes yourself. A 3-way visual merge tool such as Meld (<https://meldmerge.org/>) may be useful for making any necessary changes.  
  
Afterpay’s cartridge was developed using SFRA v5.3.0. In cases where templates, client JS, or CSS have been overridden, it may be useful to refer to that version to see the Afterpay-specific changes in the overridden file. In general, comments have been left in the files to make it easier to identify Afterpay-specific changes vs. base-SFRA content.

| Template File | Location & Purpose for Change |
| --- | --- |
| cart.isml | **cartridge/templates/default/cart/cart.isml**  Includes /js/afterpay.js and /css/afterpaystyle.js . Default error messaging on page will also display Afterpay-specific error messages. |
| cartTotals.isml | **cartridge/templates/default/cart/cartTotals.isml**  Display Afterpay messaging under cart totals on the cart page. |
| checkoutButtons.isml | **cartridge/templates/default/cart/checkoutButtons.isml**  Display Afterpay Express Checkout Button on the cart page. |
| paymentOptionsTabs.isml | **cartridge/templates/default/checkout/billing/paymentOptions/paymentOptionsTabs.isml**  Adds a payment tab for Afterpay and/or Cash App Pay |
| paymentOptionsContent.isml | **cartridge/templates/default/checkout/billing/paymentOptions/paymentOptionsContent.isml**  Include content displayed in the Afterpay and Cash App Pay payment tab. |
| paymentOptionsSummary.isml | **cartridge/templates/default/checkout/billing/paymentOptions/paymentOptionsSummary.isml**  Include content displayed in the payment summary when Afterpay is the payment method. |
| confirmationPaymentInfo.isml | **cartridge/templates/default/checkout/confirmation/confirmationPaymentInfo.isml**  Included Afterpay or Cash App Pay in the payment confirmation |
| checkout.isml | **cartridge/templates/default/checkout/checkout.isml**  Make all default SFRA checkout buttons hideable with CSS. Add the Afterpay and/or Cash App Pay checkout button. Add Afterpay Checkout Widget. Include Afterpay and Cash App Pay client-side JS and CSS. |
| scripts.isml | **cartridge/templates/default/common/scripts.isml**  Adds JS files required to load the Afterpay assets |
| pageFooter.isml | **cartridge/templates/default/components/pageFooter.isml**  Adds Afterpay among the accepted payment methods in the footer. |
| addToCartButtonExtension.isml | **cartridge/templates/default/product/components/addToCartButtonExtension.isml**  Add express checkout buttons on the PDP page. |
| setItems.isml | **cartridge/templates/default/product/components/setItems.isml** |
| bundleDetails.isml | **cartridge/templates/default/product/bundleDetails.isml**  Add Afterpay messaging for bundles. Include Afterpay JS / CSS. |
| productDetails.isml | **cartridge/templates/default/product/productDetails.isml**  Include Afterpay JS / CSS. Display Afterpay messaging on product details page. |
| productTile.isml | **cartridge/templates/default/product/productTile.isml**  Display Afterpay messaging on product tile. |
| quickView.isml | **cartridge/templates/default/product/quickView.isml**  Display Afterpay messaging on quick view.. |
| setDetails.isml | **cartridge/templates/default/product/setDetails.isml**  Include Afterpay JS/CSS |
| setQuickView.isml | **cartridge/templates/default/product/setQuickView.isml** |
| catLanding.isml | **cartridge/templates/default/rendering/category/catLanding.isml**  Include Afterpay JS/CSS |
| searchResults.isml | **cartridge/templates/default/search/searchResults.isml**  Include Afterpay JS/CSS |

Similarly, if you’ve made customizations to the following JS and SCSS files, you may need to merge the Afterpay changes.

| JS File | Location & Purpose for Change |
| --- | --- |
| checkout.js | **cartridge/client/default/js/checkout.js**  Include Afterpay client-side JS |
| productTile.js | **cartridge/client/default/js/productTile.js**  Include Afterpay client-side JS |

| SCSS File | Location |
| --- | --- |
| homePage.scss | **cartridge/client/default/scss/homePage.scss**  Adds CSS for Afterpay’s checkout button. Specifically used by the minicart when opened on the store home page. |

## Upload Changes if Necessary

If you changed any files in the previous step, either in the Afterpay cartridge, or your store’s cartridge, upload those changes to the store server.

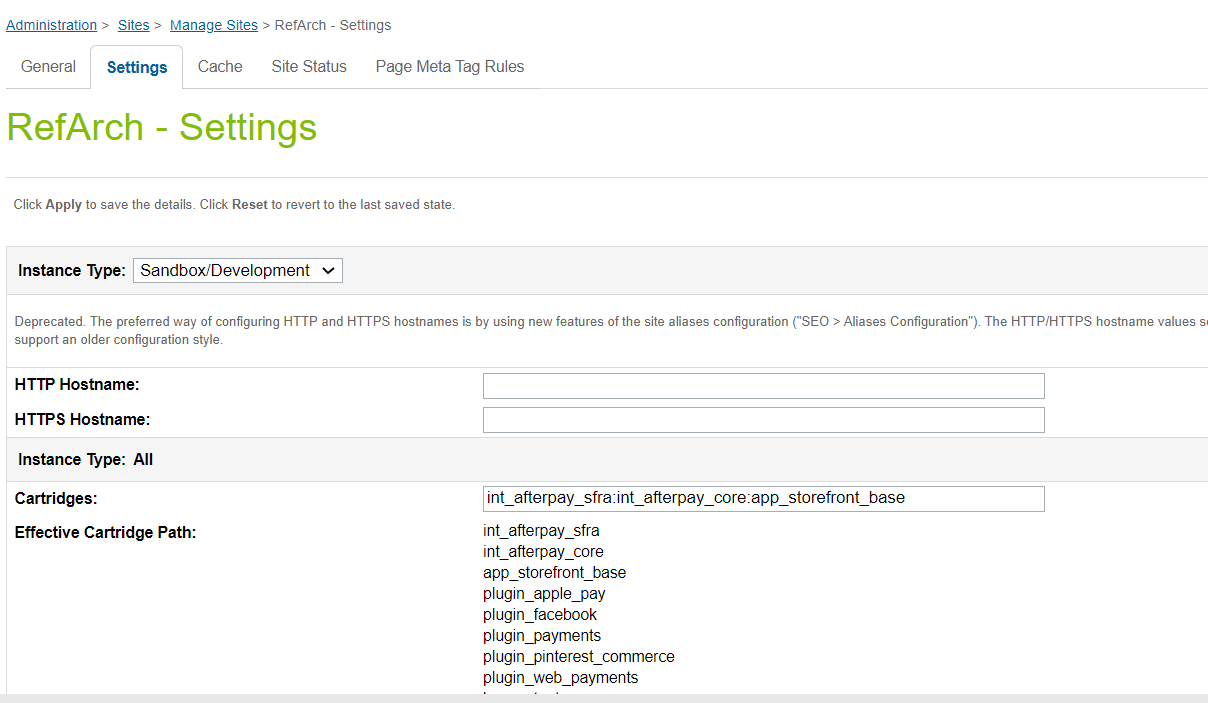
## Activate the Cartridge in Business Manager

### Set Cartridge Path

Before the Afterpay functionality can become available to SFRA, the cartridge names have to be added to the cartridge path of the Site’s settings. In order to do this, follow the below instructions:

1. Log into Business Manager
2. Navigate to Administration > Sites > Manage Sites.
3. Click on the target site name and Click on the Settings tab.
4. In the textbox Cartridges add “int\_afterpay\_sfra:int\_afterpay\_core” in front of the existing cartridge path.

If you are using SFRA 6.0.0 then you should use int\_afterpay\_sfra\_6 instead.



1. Click Apply.
2. To activate the cartridge for the Sandbox/Development/Production instances repeat steps 4 and 5 after selecting the appropriate instance from the Instance Type dropdown menu.
3. Repeat steps 3 to 6 for each site that is to use Afterpay.

### Set Afterpay Custom Site Preferences

After installing the cartridge, the following new configuration options will be available in the Business Manager.

In the Business Manager, go to Merchant Tools > Site Preferences > Custom Preferences > Integration\_Afterpay

If you currently have v21.0.0 cartridge, you will already have the express checkout options configured.

| Site Preference | Description | Default |
| --- | --- | --- |
| Enable CashAppPay  (enableCashAppPay) | The field determines if the Cash App Pay is enabled / disabled in the cartridge.  **Note : If you are setting this preference to true, please make sure:**   * **Cash App integration is actually enabled for you in the portal** * **Enable Afterpay is also true. Cash App Pay won’t load without it on the checkout page**. | False |
| Brand settings (apBrandSettings) | Afterpay / Clearpay Brand related settings  in JSON format where   * brand – afterpay/clearpay, depending on region; * service – the field is used to identify the service with Afterpay/Clearpay. * javaScriptUrl – The endpoint where the javascript library is obtained from.   Sandbox:  Afterpay (+UK) <https://portal.sandbox.afterpay.com/afterpay.js>  Clearpay <https://merchant-tools.sandbox.clearpay.com/clearpay.js>  Production:  Afterpay (+UK) <https://portal.afterpay.com/afterpay.js>  Clearpay <https://merchant-tools.clearpay.com/clearpay.js>   * learnMoreUrl – the URL that links to more information about Afterpay that users can click to view. | <default value for RefArch site. Other countries examples could be found in RefArchGlobal site meta>  {  "US": {  "brand": "afterpay",  "service": "afterpay.service.US",  "javaScriptUrl": "https://portal.sandbox.afterpay.com/afterpay.js",  "learnMoreUrl": "https://www.afterpay.com/purchase-payment-agreement"  }  } |
| Enable Express Checkout  (apEnableExpressCheckout) | Enables the Express Checkout feature. | Yes |
| Enable BuyNow  (apEnableExpressCheckoutBuyNow) | Enables the BuyNow mode. Skips review page on merchant checkout. (Note: Only used for Integrated Shipping) | Yes |
| Express Checkout Shipping Strategy  (apExpressCheckoutShippingStrategy) | Select either Integrated Shipping (Afterpay Checkout prompts for shipping method), or deferred (merchant site handles shipping method selection)  **Note: If there are multiple shipping methods in the cart, cartridge will disregard this preference and select deferred shipping strategy**. | Integrated |
| Description Express Checkout will use for In-Store Pickup  (apStorePickupDescription) | prompt to use in Afterpay Express Checkout for in-store pickup orders | Available for next-day pickup |
| Enable Express Checkout on Product Details Page  (apEnableExpressCheckoutPdp) | enable Express Checkout directly from the product detail page | No |
| Restricted Products (apRestrictedProducts) | The product ids of products that are to be excluded from Afterpay/Clearpay separated by “,” |  |

* When Afterpay is turned on, by default Express Checkout will be enabled, Integrated Shipping will be enabled, and the BuyNow option will be enabled. Make sure to change these settings to the ones you wish to use on your site.

### When Express Checkout is disabled, the Afterpay checkout button will only appear on the checkout page.

### Set Afterpay Payment Methods

Merchant Tools > Ordering > Import & Export > Upload the “payment-methods.xml” from the metadata folder > Import the xml. Payment methods with the name AFTERPAY,CLEARPAY and/or CASHAPPPAY are available that were imported.

### Set Afterpay Payment Processors

Merchant Tools > Ordering > Payment Processors > Click New Give a unique ID for the processors:

* “AFTERPAY\_CREDIT” Description: “Pay By Installment” and click Apply
* “CASHAPP\_CREDIT” Description: “Cash App Pay and” click apply

### Set SFCC(Demandware) Service

Administration > Operations > Services > Credentials > afterpay.http.defaultcredentials > Details > Credentials tab and select required service and enter the account details.

| URL | **Sandbox**  https://global-api-sandbox.afterpay.com/v2/  **Production**  https://global-api.afterpay.com/v2/ |
| --- | --- |
| User | Enter the Merchant ID provided by Afterpay |
| Password | Enter the Secret Key provided by Afterpay |

### Setting up Job Schedule

This cartridge supports an Afterpay order status of PENDING by running a Job Schedule against all Afterpay Orders that are in a PENDING state. This is not required at this point as Aferpay will only respond with APPROVED or DECLINED and if the order is DECLINED the checkout cannot complete.

By default Job Step is set for SiteGenesis site. You may change this to your site.

# Testing

## Checkout Flow

Once the cartridge is installed and integrated based on the instructions, please try to place an order on your sandbox to test the functionality. Even if you’ve enabled Express Checkout, place a non-Express-Checkout order by putting items in the cart and proceeding to the checkout page before placing an order using Afterpay.

* After you click on Place order then you should be redirected to the Afterpay Payment Page. If you are not registered then you will be asked to create an account



## Express Checkout Flow

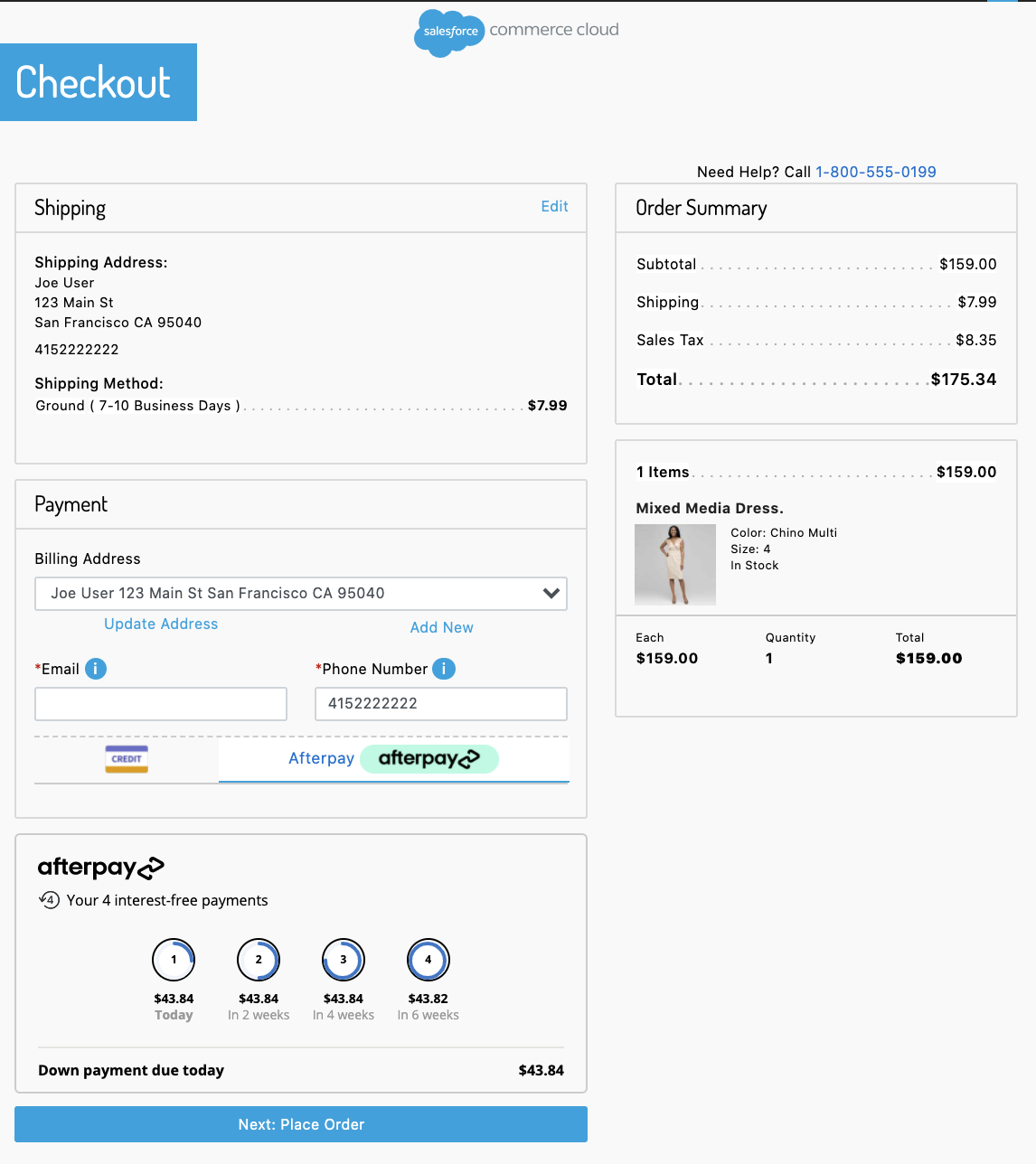
If you’ve enabled Express Checkout, please try to place an order on your sandbox to test the functionality. Use the Express Checkout buttons on the cart page and verify that Afterpay’s Express Checkout popup appears and proceed through the checkout.

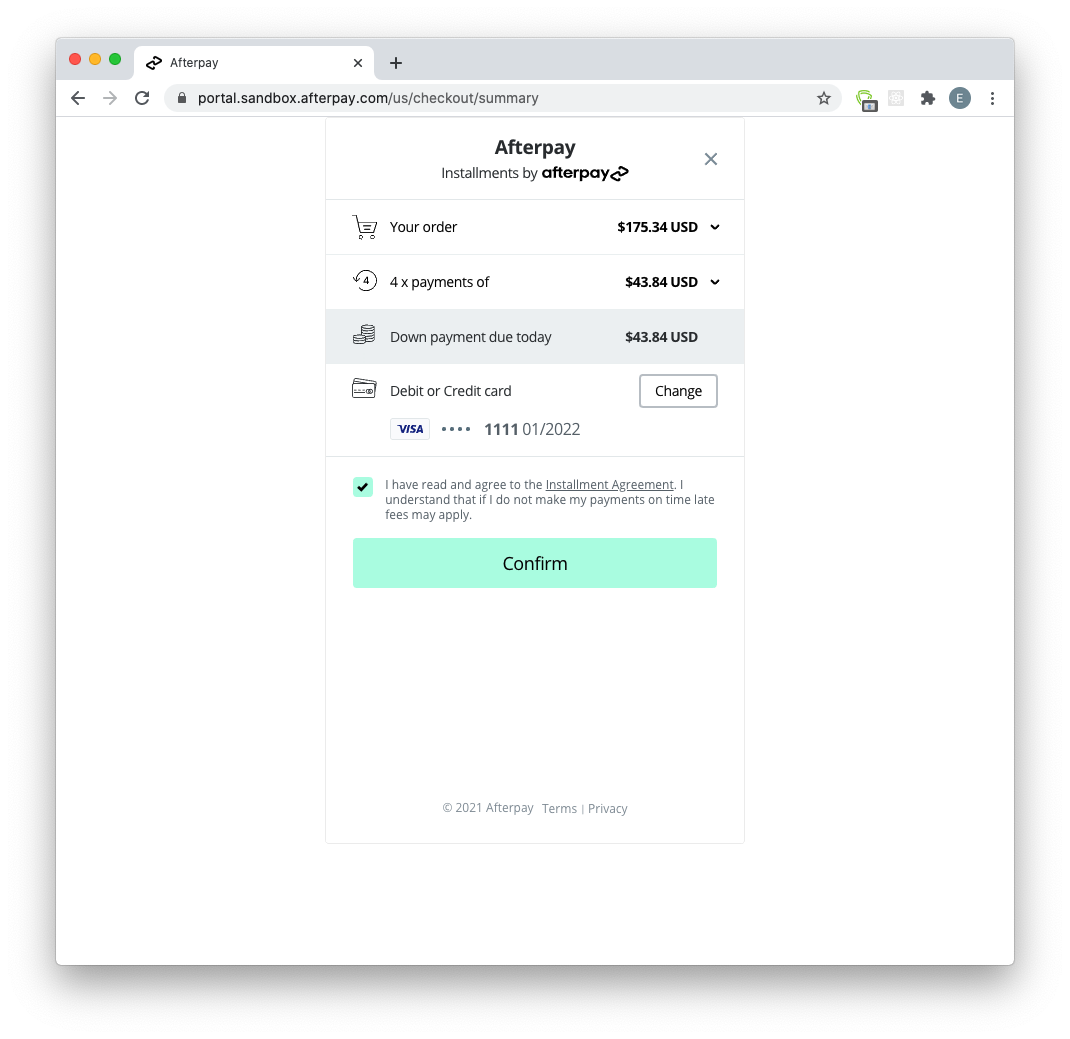
# 

# Features and Usage

## Afterpay Payment on Checkout

When the shopper selects the Afterpay payment method at checkout and clicks the **Place Order** button, they will be redirected to Afterpay servers to handle the payment. After the payment is complete, the shopper is redirected back to the confirmation or order review page on your store

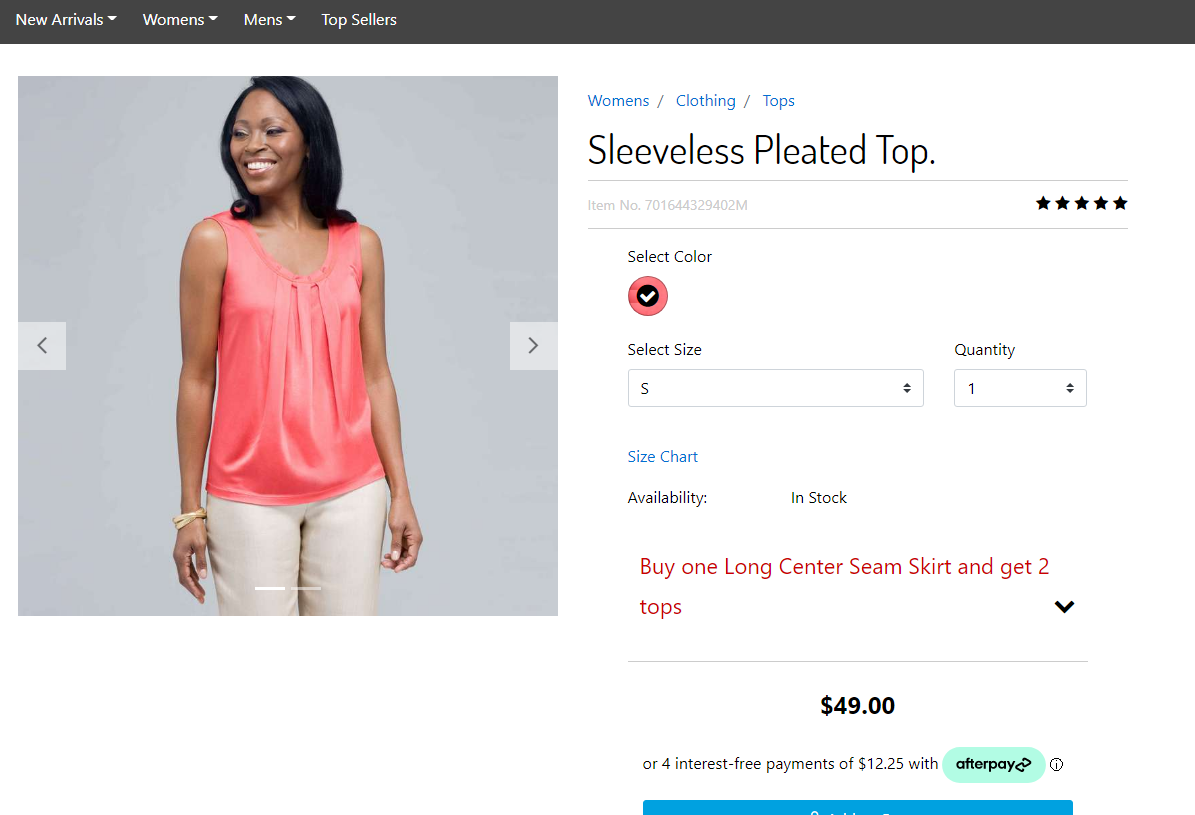




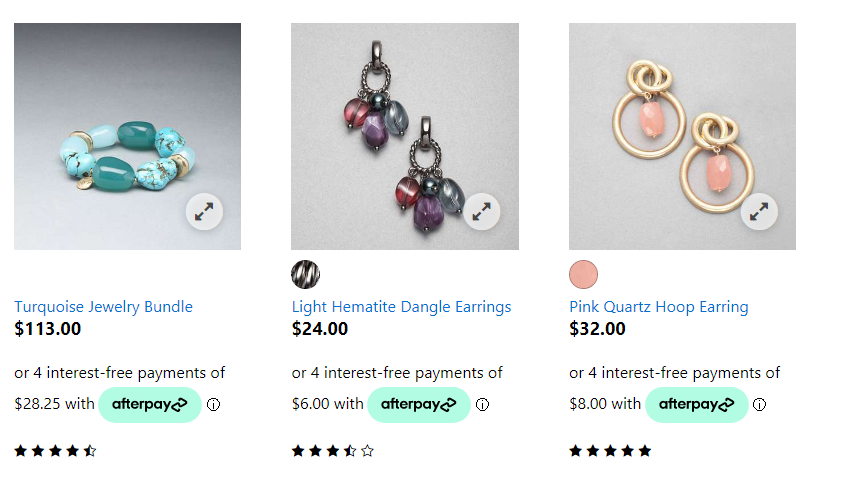
## Afterpay Product Messaging

Afterpay adds messaging on various pages in the store.

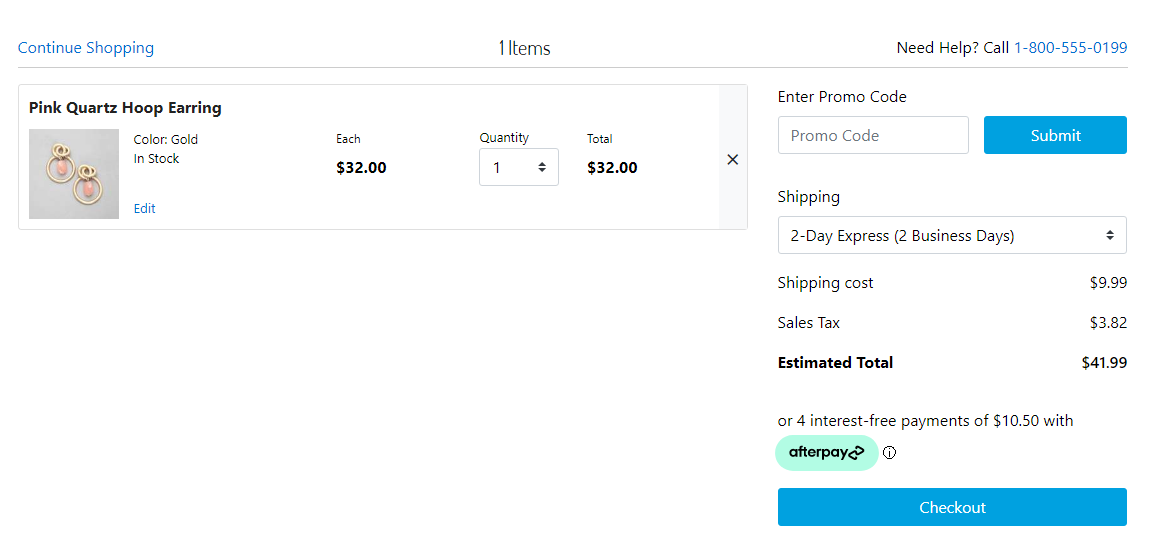
Messaging on Product Details Page:



Messaging on Product List Page:



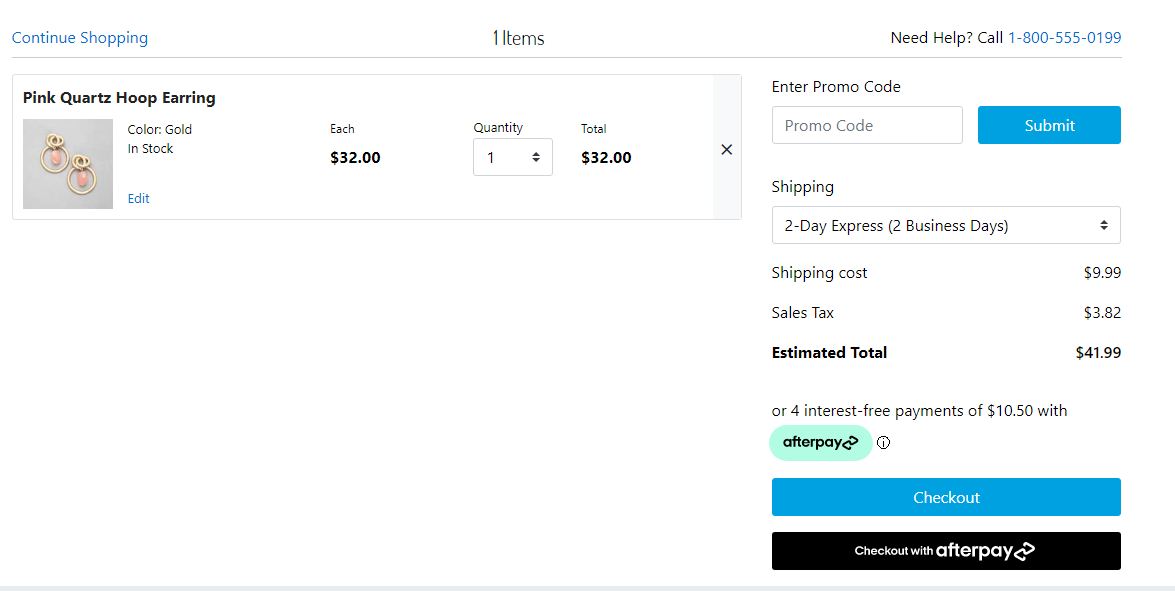
Messaging on Cart Page:

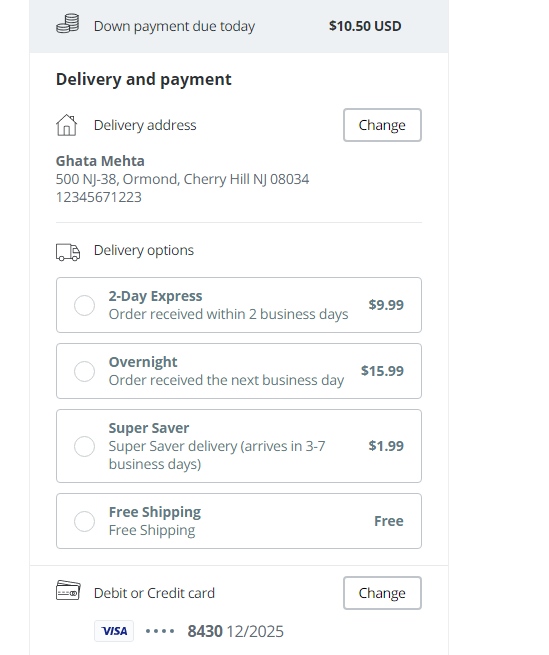


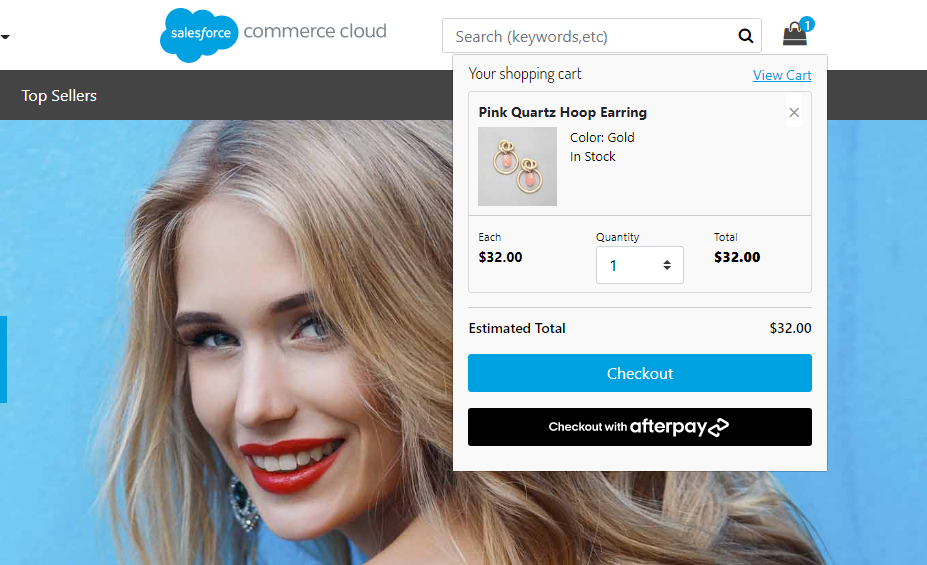
## Express Checkout

### Checkout from Cart Page

When the Express Checkout feature is enabled, new buttons will appear on the cart page and the mini cart page.



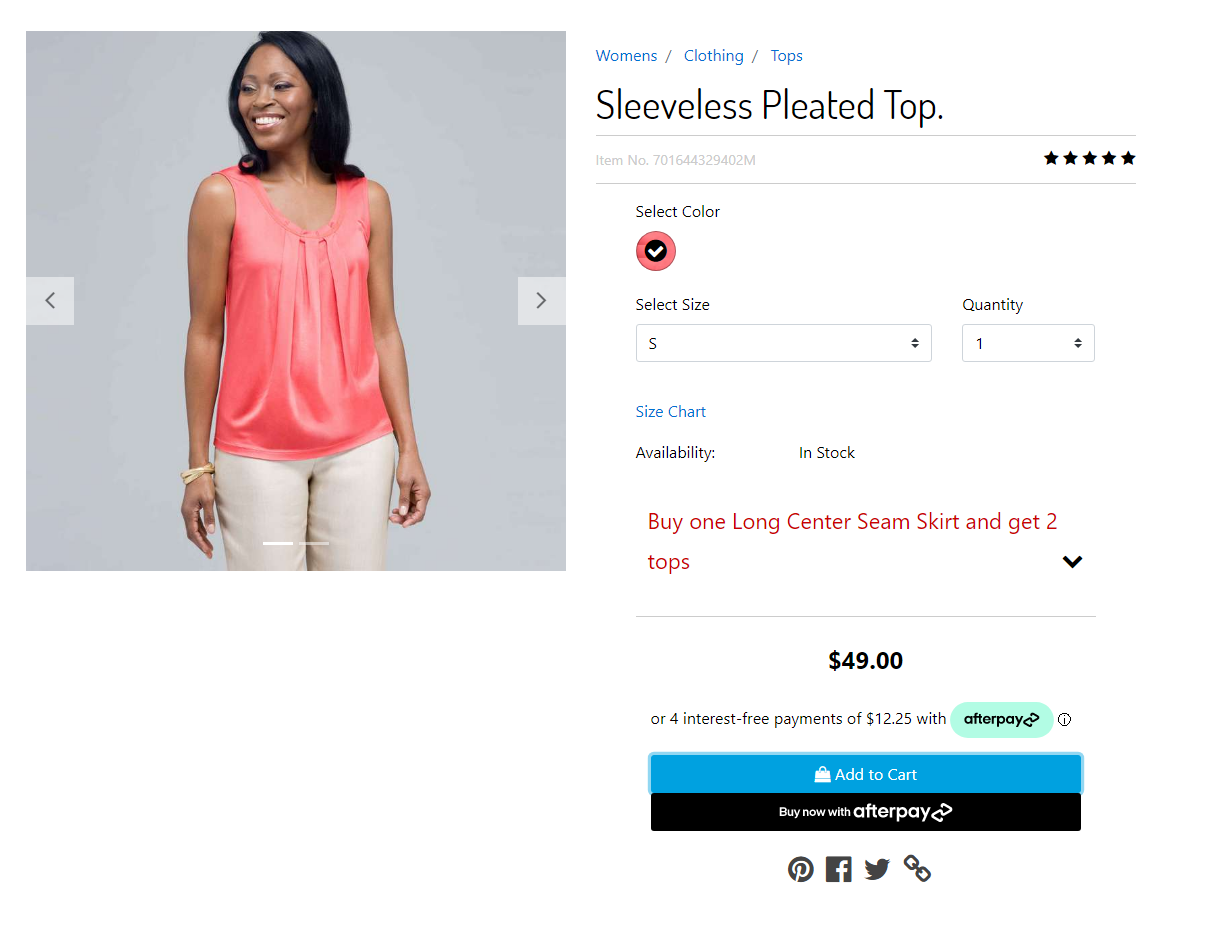




When the shopper presses either of these buttons, it will launch Afterpay Express Checkout. A popup window will appear and depending on whether the shopper has preset default checkout settings with Afterpay, they will be prompted for payment information and a shipping address. Assuming the **integrated shipping** option has been selected by the merchant, Afterpay will call a webservice, in the background, on the merchant store to get available shipping options and pricing, and display that to the shopper in the same popup window. If the BuyNow option has been selected by the merchant, the shopper can simply click “Buy Now” and will be redirected back to the merchant site, where an order will be immediately created and payment auth or capture performed (see [API Docs for Payment Flows](https://developers.afterpay.com/afterpay-online/docs/api-calls-and-payment-flows)).

### Checkout From Product Details Page

When the apEnableExpressCheckoutPdp setting is enabled, the product details page will display a “Buy now with Afterpay” button, whenever the item is eligible to be added to the cart and when the item is below the maximum threshold amount for Afterpay (typically $2000 USD).



When this button is clicked, the current item and quantity will immediately be added to the cart, and Afterpay Express Checkout will start with all the cart contents.

### Integrated Shipping

This feature improves the user experience by integrating your shipping options directly into the Afterpay Express checkout. It streamlines the checkout process and can be combined with the BuyNow option to create a one-stop checkout.  
  
To enable Integrated Shipping, in **Business** **Manager**, make sure to set the custom preference **apExpressCheckoutShippingStrategy** to **integrated**.

During Afterpay Express Checkout, once the shopper’s delivery address is known, and whenever the shopper changes their delivery address, the Afterpay popup will send a request to the merchant server to supply valid shipping options and costs for that address to the popup. The shopper makes the selection in the Afterpay popup, and the selected shipping option is passed back to the merchant site for order creation. This avoids the need for having the shopper make the shipping method selection on the merchant site.



### In-Store Pickup and Integrated Shipping

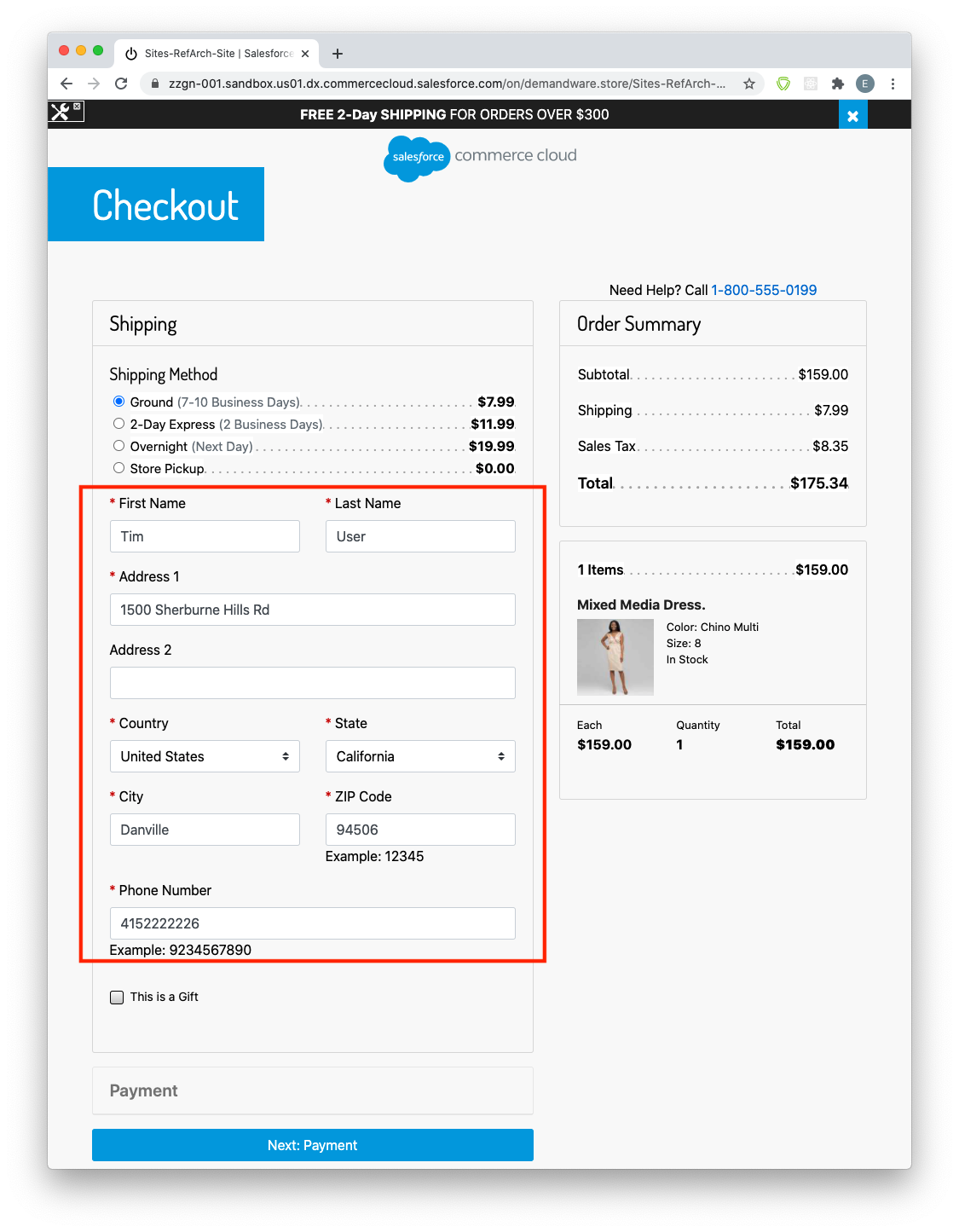
In-Store pickup is supported out of the box by **Integrated Shipping**. All items in the cart must be picked up from the same store, and there can not be a mix of in-store pickup items and home delivery items in the cart. When those conditions are met, when the Afterpay checkout button is pressed (**from the cart and mini cart page only**), the store information is passed to Afterpay in the **Create** **Token** request and the shopper will be presented with an in-store pickup interface.



If the Afterpay checkout button is pressed from the product details page, the checkout will assume the order is for home delivery.

### Deferred Shipping

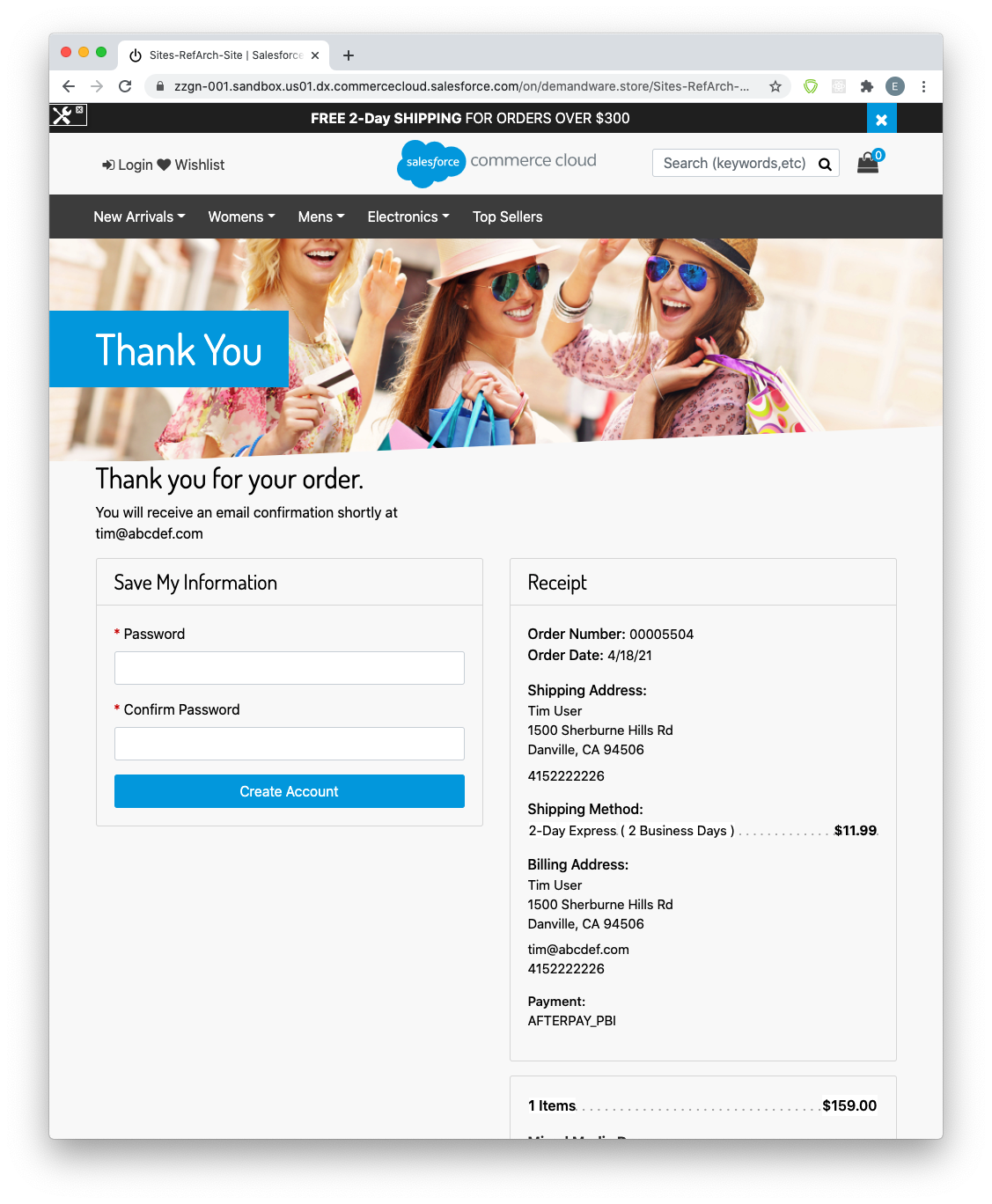
In order to handle shipping method selection on your site instead of via Afterpay’s checkout popup window, please enable the **Deferred Shipping** feature. This allows you to use the Afterpay Express Checkout to simplify the shopper’s checkout process, and quickly get the shopper’s name, email and shipping address and skip having to enter them in your checkout forms. After the shopper has gone through the Afterpay checkout process, they will be redirected back to the Shipping/Order Review page on your site to proceed with checkout.



To enable Integrated Shipping, in Business Manager, make sure to set the custom preference **apExpressCheckoutShippingStrategy** to **integrated**.

### BuyNow (Immediate capture) Feature

When the Express Checkout is complete, you may either capture payment immediately or continue the checkout on your site. If you are capturing immediately, set the **apEnableExpressCheckoutBuyNow** custom preference setting to **Yes** - this shows the user a “Buy Now” button at the end of their Afterpay checkout. Upon clicking, control is returned to your site for payment capture, and an order summary will be displayed.



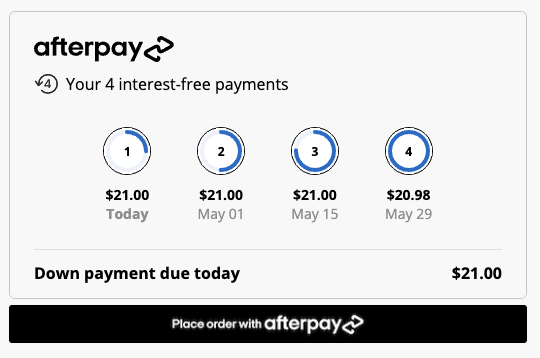
**Note**: The BuyNow feature only applies to the Integrated Shipping option. For Deferred Shipping control needs to return to your site for shipping method selection, so a BuyNow feature would not make sense.

### Afterpay Widget

The Afterpay Widget is not a new feature in and of itself, but is an essential component that needs to be placed on your site if the order value is going to change after the Afterpay Express checkout has finished. This will be the case unless the **Integrated** **Shipping** and **BuyNow** features are enabled.  
  
It serves several purposes:

1. To ensure the user has seen and agreed to an accurate payment schedule.
2. To verify that the final amount captured matches what the user has seen.
3. To provide information to the merchant such as the first payment amount for display purposes.
4. To inform the merchant of any barriers to purchase, for instance the user having gone over their payment limit with Afterpay.

The Afterpay Widget will be displayed on the final checkout summary screen (for deferred shipping):



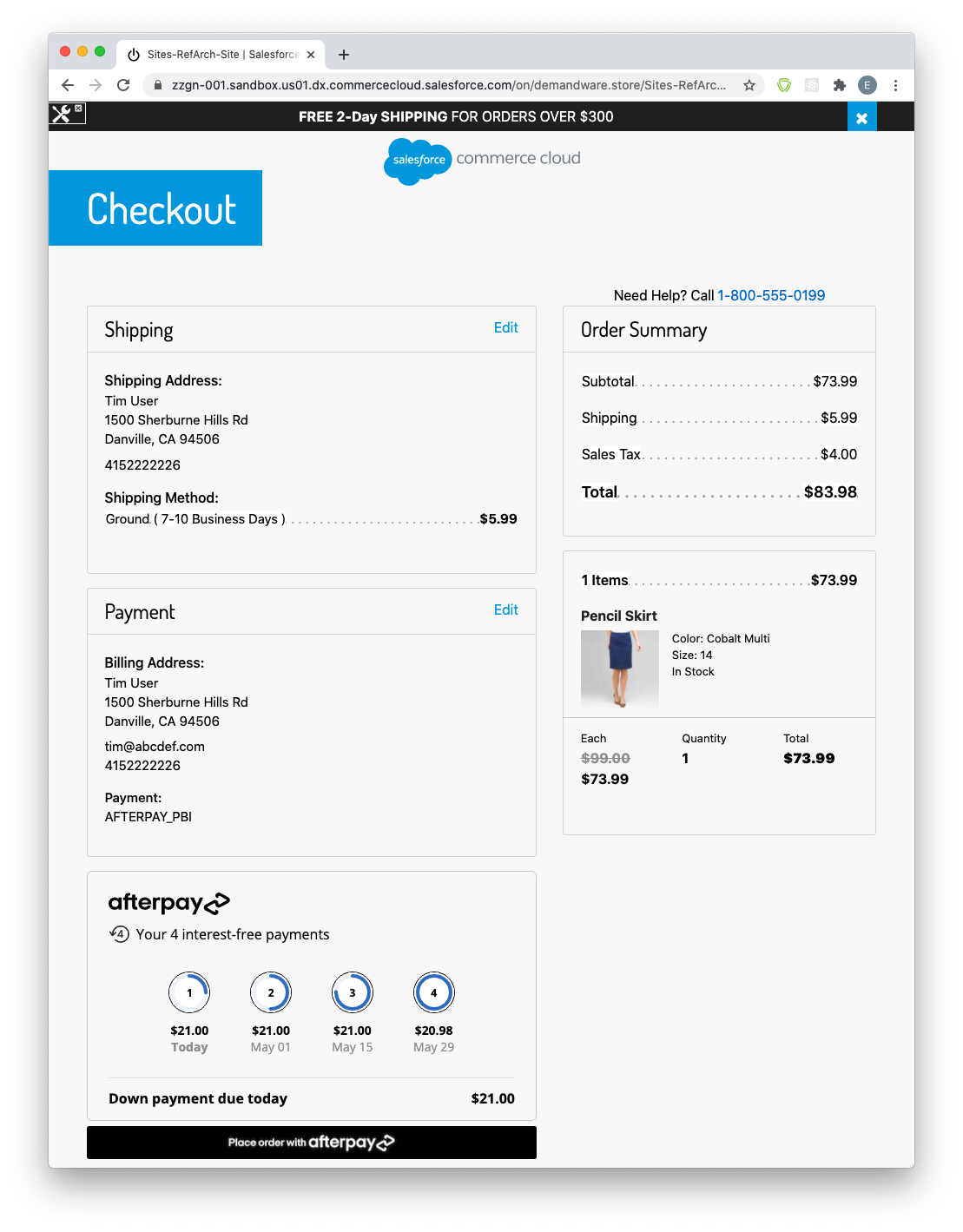
It will also be displayed on the billing screen if the shopper has already been through the Afterpay checkout process and are now back on your site to complete the checkout (see Express Checkout Finalization Flow below). It is only displayed when Afterpay is the active payment instrument for the cart.

The **Place order with Afterpay** button will always be placed below the Afterpay Widget to ensure that the shopper has seen the widget before placing the order.

### Express Checkout Finalization Flow (ECFF)

Unless you are using the **Integrated** **Shipping** feature with the **BuyNow** option, the shopper will be redirected back to your site after the Afterpay checkout. During this phase, in SFRA, the shopper can still make arbitrary changes to their cart (e.g. add/remove items, apply coupons and gift certificates, change shipping address and methods, etc) . However, Afterpay’s link cartridge will remember (using session state) that the user is currently in the Express Checkout Finalization Flow. This will affect the behavior of the Afterpay checkout buttons:

* When the “Checkout with Afterpay” button is clicked, instead of redirecting to Afterpay to checkout again and generate a new token, the button will take the shopper to the cart summary page, where the Afterpay Widget will be displayed, along with the “Place Order with Afterpay” button.



* When the “Buy now with Afterpay” button is clicked on the product detail page, the product will be added to the cart and the shopper will be directed to the summary page as well. The default shipping method used in the rest of the order will be used for the newly added product as well. The Afterpay Widget will display the total order amount, and upon clicking the “Place Order with Afterpay” button, the order will be created and payment will be captured.

### Exiting from Express Checkout Finalization Flow:

There are several conditions which will cause the shopper to exit from ECFF:

* If the user explicitly chooses another payment method on the billing page, and clicks submit, Afterpay will be removed as a payment method.
* If the user navigates to any page which normally contains an Afterpay Widget (the billing page and summary page), but does not have Afterpay set as a payment instrument for the cart, they will no longer be in ECFF.
* After the shopper clicks **Place Order with Afterpay**, regardless of whether the transaction was successful or not, they will no longer be in ECFF. The **Place Order with Afterpay** button will also fail if the user is not in “Express Checkout Finalization Flow”, but this should not happen unless the shopper opens multiple browser windows and clicks on the button in an old window when the session state has already changed.
* The user’s browser session expired

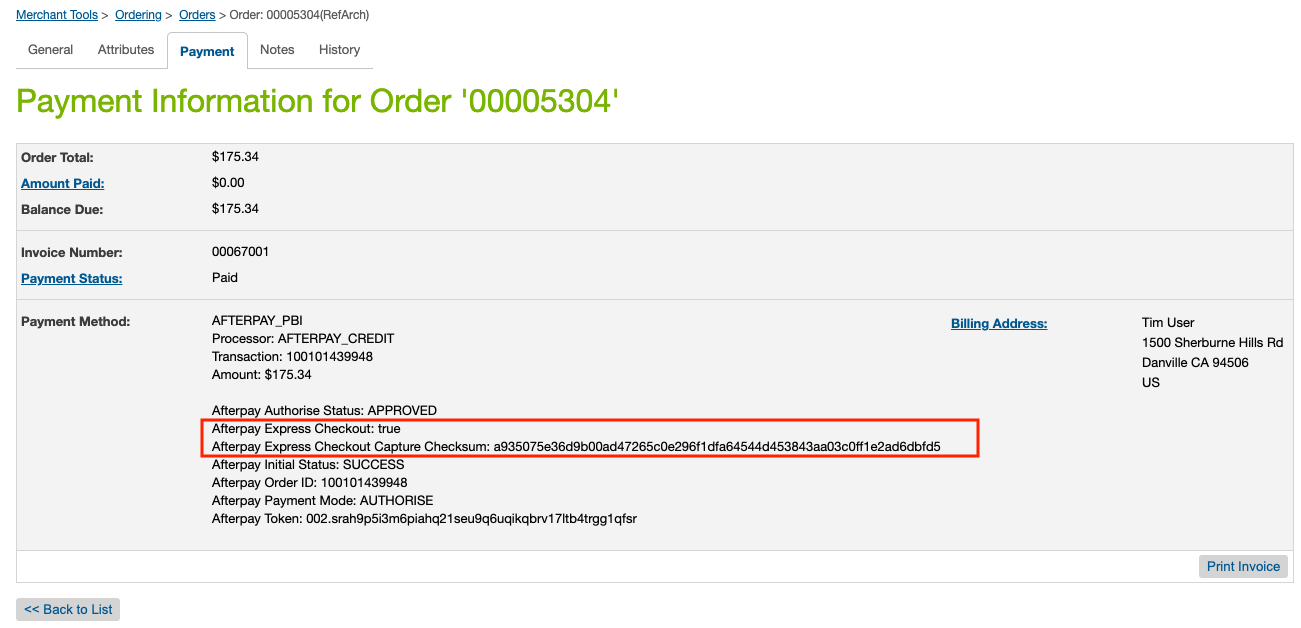
After exiting from the Express Checkout FInalization Flow, clicking on “Checkout with Afterpay”, or “Buy Now with Afterpay” buttons will start the Afterpay checkout process again.

### Express Checkout Behavior

### Order Fields

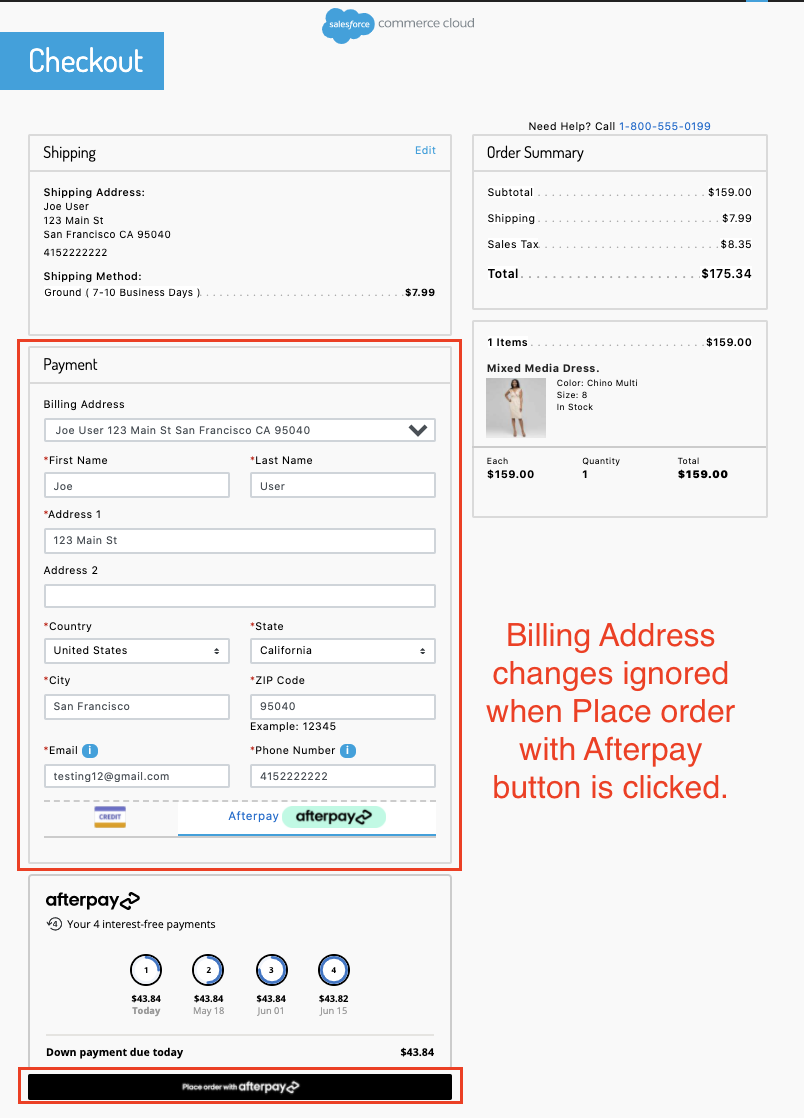
Orders created via Afterpay Express Checkout will look the same as non-Express Checkout orders, with the following exceptions:

* The Billing Address field will display the shipping address. Since Afterpay is handling the checkout, after the express checkout popup closes, the shipping address is returned from Afterpay and used to populate both the shipping address and billing address in the cart. Thus, when the order is created, it will be the shipping address unless the shopper has specifically modified the billing address before finalizing the transaction.
* **Afterpay Express Checkout** - will be true for express checkout orders
* **Afterpay Express Checkout Capture Checksum** - this is a checksum generated by the Afterpay Widget, and indicates the total amount and payment schedule the shopper was shown before placing the order



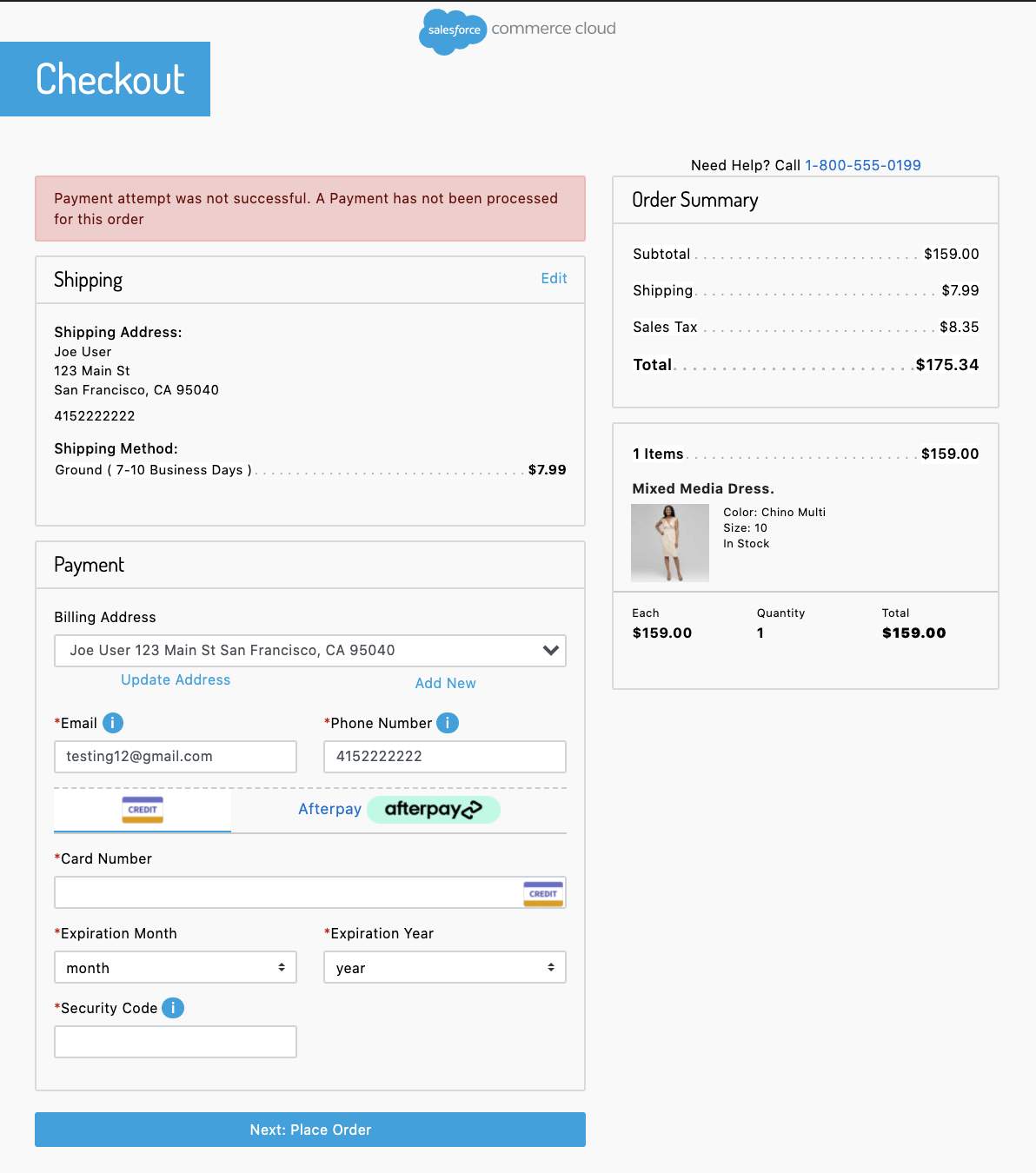
### Billing Page Address Changes During ECFF

On the Billing Page, if the billing address is changed right before the shopper clicks “Place order with Afterpay” (during ECFF), the changed data will be ignored since the form is not actually submitted to the server. This should be unimportant since Afterpay is handling the payment process and the billing address fields would typically be prepopulated with the shipping address.



### Error Behavior

When there are payment-related errors, the shopper will be redirected back to the billing page and an error message will be displayed.



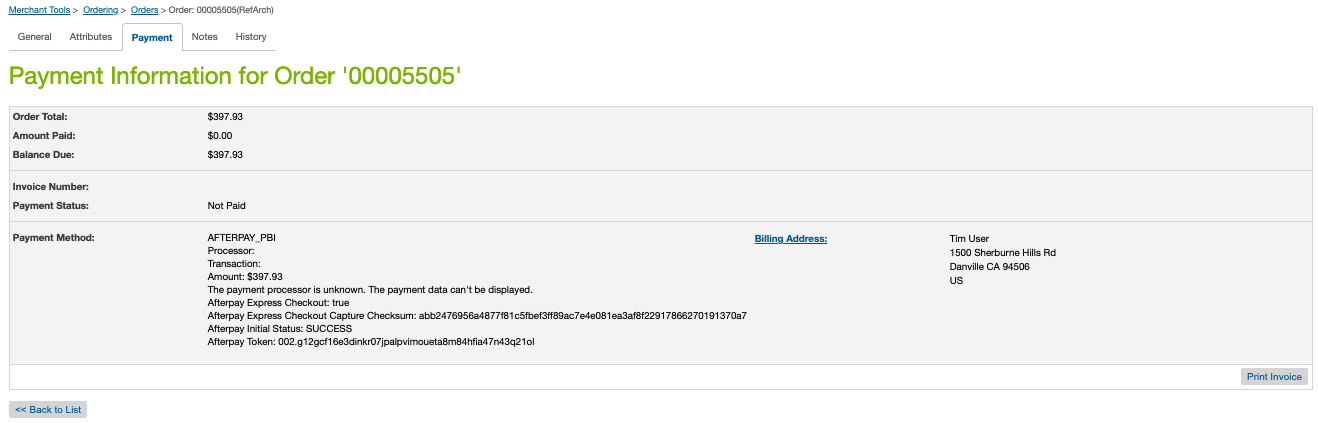
Note that if the shopper was in Express Checkout Finalization Flow, they will no longer be after a payment-related error.

When there is a payment-related error, an order will generally already have been created by SFCC. This can occur in the following scenarios:

* Capture was declined
* Shopper modified the cart in the short period between when SFCC created the order, and before the payment capture via Afterpay.
* The merchant site was unable to communicate with Afterpay’s payment servers

The order will have a “Failed” status in these cases, and the payment status will be “Not Paid”.





### ECFF Behavior

Because the shopper can make arbitrary changes to their billing address and shipping address during ECFF, occasionally the shipping and billing address on-file at Afterpay may not match the information created in the order. However, this should not matter since you (the merchant), will always have the correct shipping address, and Afterpay will always have the correct billing address.

### Express Checkout Limitations

### No Multiple Shipping Destinations Prior to Afterpay Express Checkout

Express Checkout will only work when all items in the cart are set for **Home Delivery** OR all items in the cart are set for **In-Store** **Pickup** from the same store. If neither of these are true, and the shopper clicks the “Checkout with Afterpay” button, an error message will appear indicating that Afterpay is unavailable for the current cart.



The shopper can still pay with Afterpay using the standard checkout process (selecting Afterpay on the billing page), but will not be able to use express checkout features.

Even though the shopper will not be able to use Express Checkout in these instances, the exception is if they’ve already gone through the Afterpay Express Checkout, and are now in the ECFF on your site. They will be able to add items to cart, ship to multiple locations, and/or pick up from multiple stores. The order can still be created and payment captured once they click on the “Pay now with Afterpay” button. At this point, since Afterpay is only capturing payment, the shipping limitations no longer matter.

### Billing Address is not Passed to Merchant

If a shopper uses Afterpay Express Checkout, the billing address is typically not passed back to your site from Afterpay. In non-BuyNow checkout flows, the billing address will instead be populated with the shipping address.

### Other Limitations

* Currently, there is no support for a “Buy Now with Afterpay” from a wishlist page.
* Currently, there is no support for a “Buy Now with Afterpay” button on a product bundle page.
* For in-store pickup orders, Afterpay Express checkout does not support adding a message to the store. If finalization happens at the merchant site, the shopper could go back to the shipping screen and add the message.

### Express Checkout Implementation Details

### Overview

Particularly for sites that have made substantive changes to the base SFRA store, we’ve provided some details on how certain features were implemented. This would make it easier to understand how your site’s customizations may potentially require some modifications to Afterpay’s cartridge to work properly.

### Checkout Button Behavior during ECFF

In the isml templates, buttons which behave differently during ECFF will have an <isif condition=”${isExpressCheckoutFinalize}”> to control the button behavior. In **int\_afterpay\_sfra/cartridge/templates/default/cart/checkoutButtons.isml**, during ECFF, a button will be displayed that upon clicking, will run the “AfterpayExpress-ContinueFinalize'' endpoint. This is an endpoint that will navigate to the cart summary page. When not in ECFF, the button has the (id=”afterpay-express-button”), which has been set as the target for launching the Afterpay Express Checkout popup.

If your site is designed in such a way that ECFF wouldn’t make sense and you’d prefer the user go through Express Checkout again whenever they click the “Checkout with Afterpay” or “Buy now with Afterpay” buttons, this is behavior that’s easily changeable in the template.



### Client-Side Javascript for Express Checkout

In order to support Express Checkout in the SFRA checkout flow, we run client-side JavaScript to determine when to hide/show the Express Checkout Widget and the Place Order with Afterpay buttons. Most of this logic is located in:

customCheckout.js

| // Call handleStage when page is loaded/reloaded  handleStateChange();   let cmElem = document.querySelector('#checkout-main');   // Call handleStage with new stage whenever we detect the stage change  // SFRA base changes attributes on #checkout-main to indicate stage of checkout flow changes  if (window.MutationObserver) {  var observer = new MutationObserver(function (mutations) {  for (let mutation of mutations) {  if (mutation.type === 'attributes') {  handleStateChange();  }  }  });  observer.observe(cmElem, { attributes: true });  } else {  // If no MutationObserver support, just use interval to poll state  var checkState = setInterval(function () {  handleStateChange();  }, 500);  }   // Call handleStage with new stage whenever afterpay payment tab is pressed  let tabelem = document.querySelector('.afterpay-tab') ? document.querySelector('.afterpay-tab') : document.querySelector('.clearpay-tab');  if (window.MutationObserver) {  var observer = new MutationObserver(function (mutations) {  handleStateChange();  });  observer.observe(tabelem, { attributes: true });  } |
| --- |
|  |

We use 2 MutationObservers that look for attribute changes on the ‘#checkout-main’ - the attributes on this element change as the checkout progresses through the stages of shipping / payment / placeOrder / submitted, so we use that as well to determine if the stage changed. The other MutationObserver checks if the Afterpay payments tab has been clicked.

Individual elements then use the following css classes to indicate whether the element should be shown in a certain state.

| **.ap-checkout-ship** - only shown during shipping stage  **.ap-checkout-pay-tab-ecf** - only shown during payment stage **if** Afterpay tab is active and shopper is **in** Express Checkout Finalization Flow  **.ap-checkout-pay-tab-noecf** - only shown during payment stage **if** Afterpay tab is active and shopper is NOT **in** Express Checkout Finalization Flow  **.ap-checkout-pay-notab-ecf** - only shown during payment stage **if** Afterpay tab is NOT active and shopper is **in** Express Checkout Finalization Flow  **.ap-checkout-pay-notab-noecf** - only shown during payment stage **if** Afterpay tab is NOT active and shopper is NOT **in** Express Checkout Finalization Flow  **.ap-checkout-po-ecf** - only shown during checkout stage during Express Checkout Finalization Flow  **.ap-checkout-po-noecf** - only shown during checkout stage **if** shopper is NOT **in** Express Checkout Finalization Flow |
| --- |

If you’ve customized the checkout UI for your site, please make sure there are no conflicts between this and your customizations.

### Add To Cart from PDP Page

When the shopper clicks the Buy now with Afterpay button on a PDP page, the item is added to the cart and the Express Checkout widget is displayed. In order to make this feature as flexible and easy to integrate as possible, the button acts as a normal Add to Cart button of the store, but has an additional onclick handler which starts the Express Checkout widget and waits 200ms before running the CreateToken controller endpoint on the server. The 200ms delay ensures that the item has been added to the basket.

In order to achieve this, we call initAfterpay and specify a commenceDelay value of 200.

In afterpay.js:

| **if** ($('#afterpay-express-pdp-button').length > 0) {  initAfterpay({pickupflag: "false", commenceDelay: 200, target: '#afterpay-express-pdp-button'}); } |
| --- |

This makes the Express Checkout widget wait 200ms before running the CreateToken controller route.

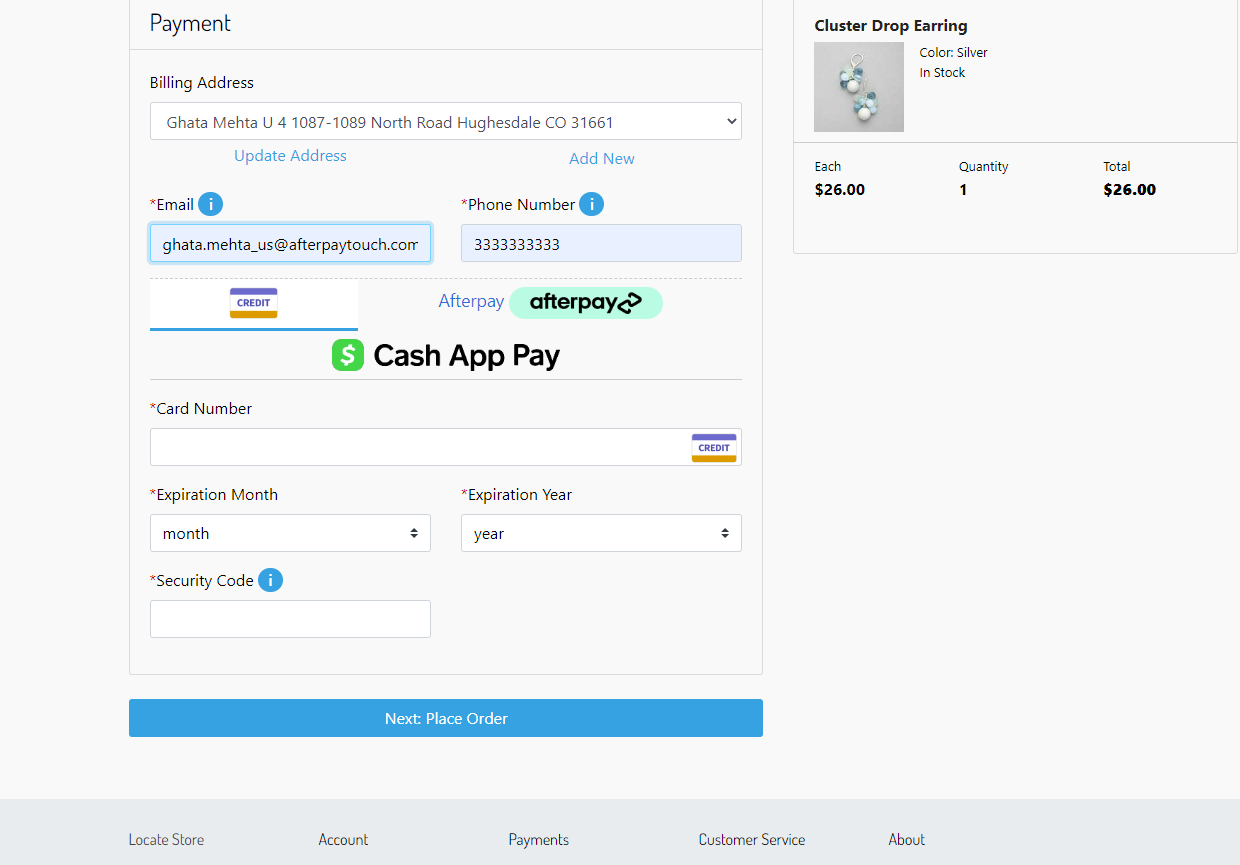
# 

# CASH APP PAY

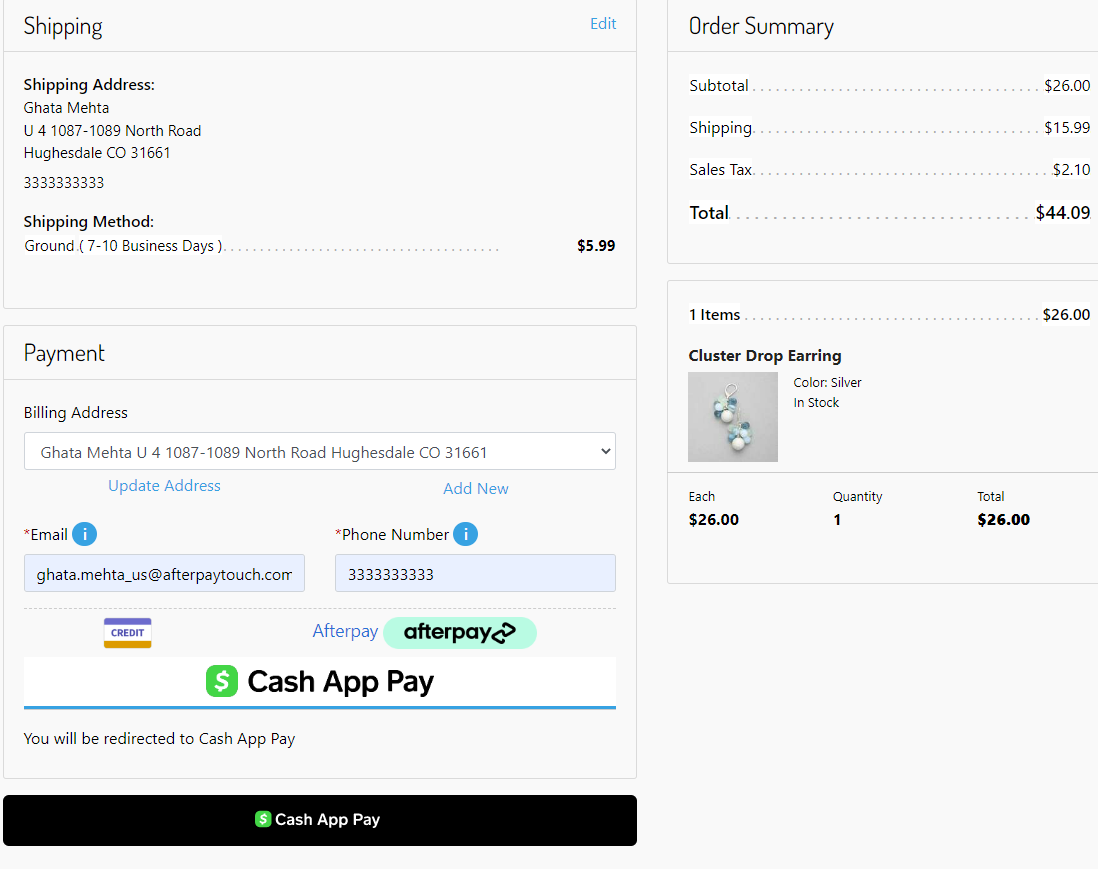
## Cash App Pay Checkout Flow

If you’ve enabled Cash App Pay, you would be able to see the Cash App Pay payment option on the checkout page. A consumer needs to have a CashApp application installed on their phone that scans the QR code on the popup..  
  
  
Checkout Page

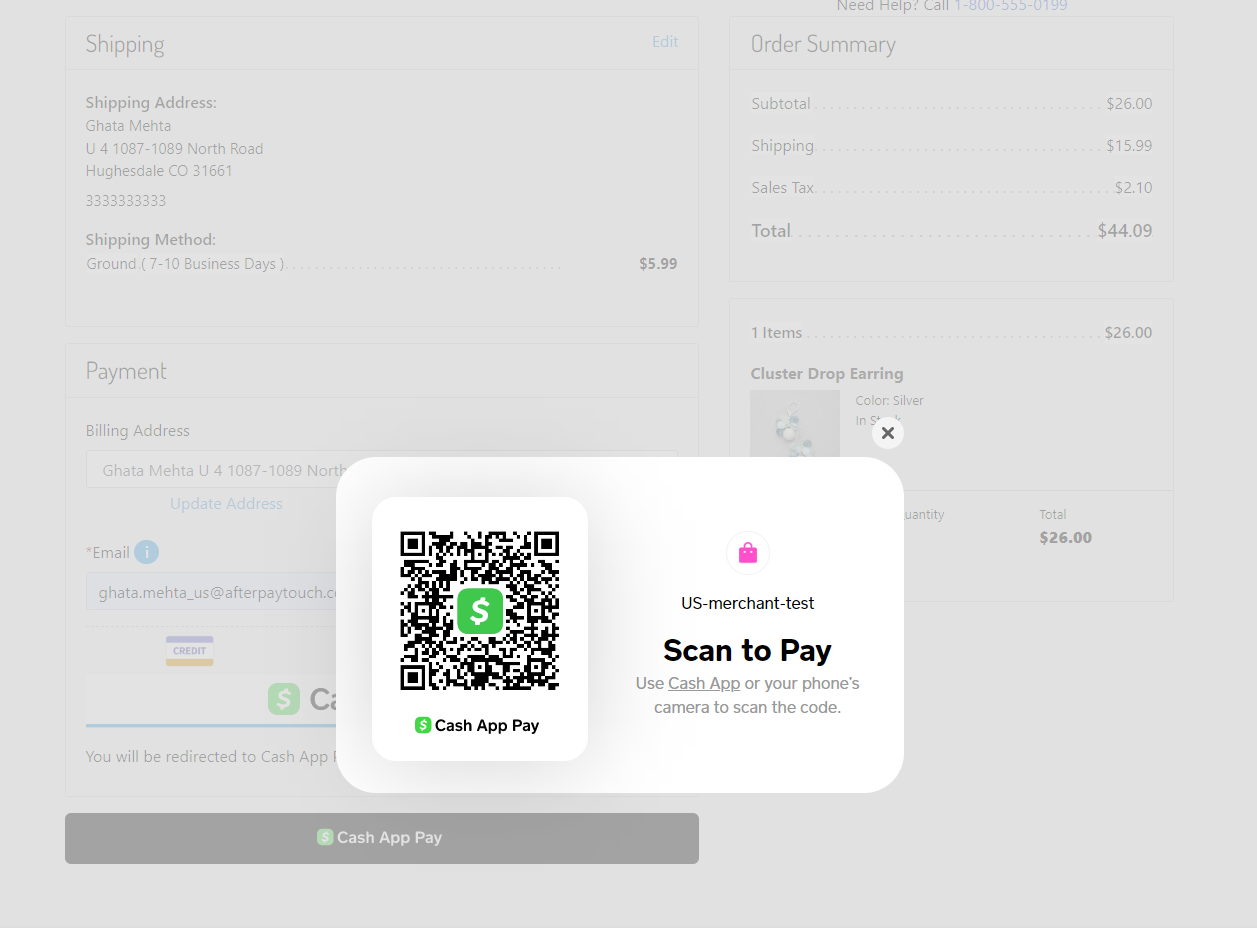
For an eligible order, Cash App Pay will appear as a payment option on the Checkout.



When you click on the Cash App Pay, a “Csh App Pay” button will be rendered.



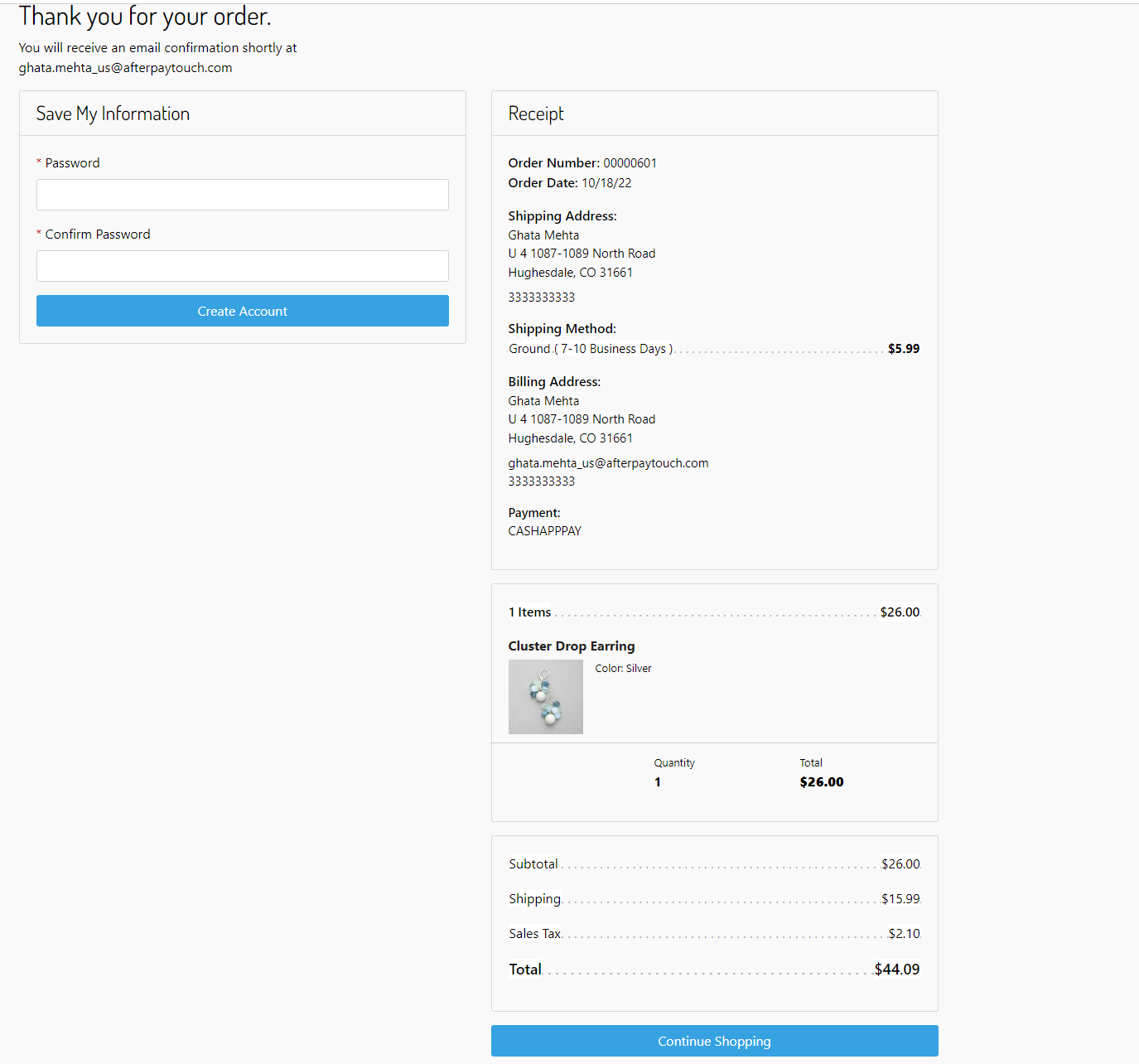
When you click on the “Cash App Pay” button, a popup will open with a QR code.



On your phone, you scan the QR code with the Cash App Application, when prompted you either accept/decline the payment.

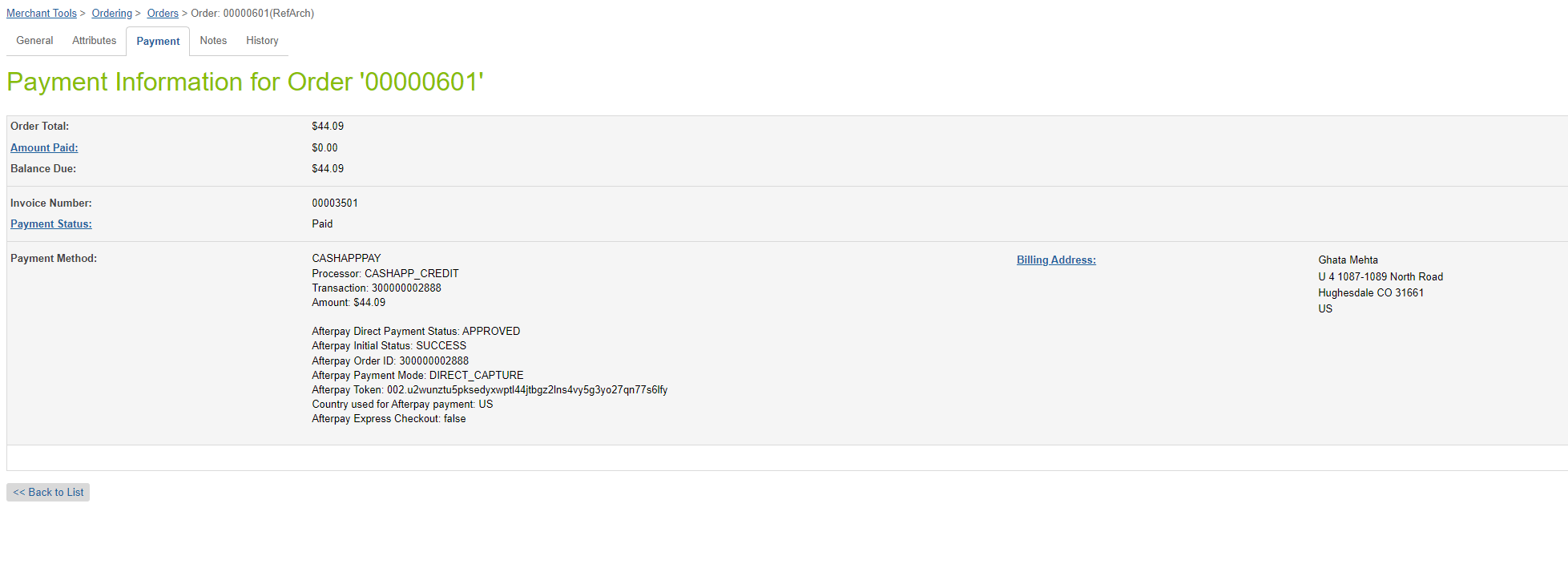


On Accepting, Cash App Pay will process the payment and you will be redirected to the order confirmation page.



Order in Business Manager

You can see the Order in Merchant Tools > Ordering > Orders



# CROSS BORDER TRADE

In v23.3.0, we have introduced support for the Cross Border trade for the eligible merchants. If the merchant has enabled cross border trade then now they can offer Afterpay/Clearpay in the eligible countries on their website.  
  
Just make sure, you have assigned correct service-id to the country code in the **“Brand settings”** field under Merchant Tools > Site Preferences > **Custom Preferences**

| {  "GB": {  "brand": "clearpay",  "service": "clearpay.service.GB.v2",  "javaScriptUrl": "https://portal.sandbox.clearpay.co.uk/afterpay.js",  "learnMoreUrl": "https://www.clearpay.co.uk/en-GB/terms-of-service"  },  "AU": {  "brand": "afterpay",  "service": "afterpay.service.AU.v2",  "javaScriptUrl": "https://portal.sandbox.afterpay.com/afterpay.js",  "learnMoreUrl": "https://www.afterpay.com/en-AU/terms-of-service"  },  "NZ": {  "brand": "afterpay",  "service": "afterpay.service.NZ.v2",  "javaScriptUrl": "https://portal.sandbox.afterpay.com/afterpay.js",  "learnMoreUrl": "https://www.afterpay.com/en-NZ/terms-of-service"  },  "CA": {  "brand": "afterpay",  "service": "afterpay.service.CA.v2",  "javaScriptUrl": "https://portal.sandbox.afterpay.com/afterpay.js",  "learnMoreUrl": "https://www.afterpay.com/en-CA/instalment-agreement"  } } |
| --- |
|  |

# Operations / Maintenance

## Data Storage

Some additional custom attributes are stored against system objects. The following table lists these attributes and their uses, other than Site Preferences (which are listed under section 3.3.2).

| Object | Attribute | Type | Description |
| --- | --- | --- | --- |
| OrganizationPreferences | flowIPAddresses (flowIPAddresses) | Text | IP addresses |
| Order | Afterpay Payment Method (apIsAfterpayOrder) | Boolean | Flags the order as an Afterpay order |
| Order | Cash App Pay Payment Method  (isCashAppPayOrder) | Boolean | Flags the order as an Cash App Pay order |
| PaymentTransaction | Afterpay Token (apToken) | String | The token used to place order in Afterpay |
| PaymentTransaction | Afterpay Initial Status  (apInitialStatus) | String | The pre-approval status first returned |
| PaymentTransaction | Afterpay Payment Mode (apPaymentMode) | String | The payment mode used |
| PaymentTransaction | Afterpay Direct Payment  (apDirectPayment Status) | String | The status of Direct Payment, if this payment mode is used |
| PaymentTransaction | Afterpay Authorise Status  (apAuthoriseStatus ) | String | The status of Authorise, if  this payment mode is used |
| PaymentTransaction | Afterpay Order ID (apPaymentD) | String | The Payment ID generated by Afterpay |
| PaymentTransaction | Afterpay Refund ID (apRefundID) | String | Refund ID |
| PaymentTransaction | Auth Expiry (apAuthExpiry) | String | The time the Afterpay Order Auth Expires |
| PaymentTransaction | Payment Event ID (apPaymentEventID ) | String | The unique ID for the Afterpay Auth event |
| PaymentTransaction | Afterpay Express Checkout (apExpressCheckout) | Boolean | Flags the order as an express checkout order |
| PaymentTransaction | Afterpay Express Checkout Checksum (apExpressCheckoutChecksum) | String | Checksum from the Afterpay Widget displayed before the order was created. |

## Availability

If the Afterpay service is unavailable, the shopper will not be able to checkout using this payment method.

The service availability can be tracked in SFCC using **Administration** **>** **Operations** **>** **Service** **Status**.

## Support

For support, please contact your Afterpay account contact.

# 

# User Guide

## Roles, Responsibilities

Integration of this cartridge will typically be done by a SFCC developer.

Afterpay will provide access keys for be used with the API.

## Business Manager Refund Transaction

For the Afterpay integration to work, the following needs to be configured in the Business manager.

1. Upload the cartridge bm\_afterpay and int\_afterpay\_core into the server.
2. Go to Business Manager Menu > Administration > Manage Sites > Business Manager > Settings > Cartridges input field add cartridge name “bm\_afterpay: int\_afterpay\_core.
3. Administration > Organization > Roles > Administrator > Business manager modules > Select context (select your site) > enable the Afterpay transactions management

# 

# Release History

| Version | Date | Changes |
| --- | --- | --- |
| 23.3.0 | 11 July 2023 | CBT, Restricted products |
| 23.2.0 | 2 Mar 2023 | Cash App Pay for US, Minor bug fixes |
| 23.1.0 | 28 Feb 2022 | Support split shipment, Support for SFRA 6.0 |
| 23.0.1 | 18 Nov 2021 | Security improvements, Minor bug fixes |
| 23.0.0 | 15 Sep 2021 | Afterpay JS library, Multilingual |
| 21.1.0 | 30 Apr 2021 | Express Checkout, V2 API |
| 20.1.0 | 10 Jan 2020 | No changes |
| 18.3 | 07 May 2018 | Controller version |
| 18.3 | 03 Sep 2016 | Initial release |