

ParadoxLabs Stripe: User Manual

Version 5.0 – For Magento 2.4.4+ – Updated 2026-07-17

Table of Contents

ParadoxLabs Stripe: User Manual.....	1
Installation.....	3
If you purchased from Magento Marketplace	3
If you purchased from store.paradoxlabs.com	4
Updating Stripe.....	5
If you purchased from Magento Marketplace	5
If you purchased from store.paradoxlabs.com	5
Configuration	6
General.....	6
Checkout Settings	7
Payment Settings	8
Advanced Settings.....	9
Behavior Notes	10
Usage	12
Checkout Payment Form.....	12
Order status page	14
Customer ‘My Payment Options’ account area	15
Admin order form	16
Admin order status page.....	16
Admin customer ‘Payment Options’ account area	17
Admin transaction info	17
Frequently Asked Questions & Troubleshooting.....	19
Is ParadoxLabs Stripe PCI Compliance?	19
Does ParadoxLabs Stripe support SCA?	19
How do I do an online refund from Magento?	19
Do you support Apple Pay?.....	20
How does this payment method handle currency?	20
Technical / Integration Details.....	21

Architecture	21
Custom customer attributes	21
Custom database schema	21
Events.....	21
Magento API: REST and SOAP	22
Magento API: GraphQL	32
Split Database	40
Support	41

Installation

The installation process differs based on where you purchased our extension.

If you purchased from Magento Marketplace

NOTE: You will not be able to install by downloading the extension files from Marketplace.

The Marketplace download does not include all of the necessary files. You must install using either the Web Setup Wizard or Composer, with the following directions.

Step 1: Install

We strongly recommend installing, configuring, and testing all extensions on a development website before installing and using them in production.

If you encounter any problems during this process, please contact [Magento Marketplace Support](#).

Note, installing this extension requires familiarity with your server's command line. Ensure your server has composer set up and linked to your Magento Marketplace account (including repository <https://repo.magento.com>). Then in SSH, from your site root, run the following commands:

```
composer require paradoxlabs/stripe:*
php bin/magento module:enable -c ParadoxLabs-TokenBase ParadoxLabs_Strip
php bin/magento setup:upgrade
```

If your site is in production mode, you will also need to run these commands to recompile sources:

```
php bin/magento setup:di:compile
php bin/magento setup:static-content:deploy
```

These commands should load and install the extension packages from the Marketplace repository.

Composer installation is only available for Marketplace purchases.

Step 2: Configure

See the configuration section below.

If you purchased from store.paradoxlabs.com

NOTE: This file upload installation applies **only** to purchases from the ParadoxLabs Store. Marketplace purchases must follow the Marketplace installation directions above.

Step 1: Upload files

Upload all files within the **upload** folder into the root directory of Magento.

Folder in Download	Folder on Server
/upload/app/	→ /app/

Step 2: Run Installation

In SSH, from your site root, run the following commands:

```
composer require "stripe/stripe-php:^20.0"
php bin/magento module:enable -c ParadoxLabs-TokenBase ParadoxLabs_Stripe
php bin/magento setup:upgrade
```

It is very important that you run all three lines above in that exact order.

These will install the Stripe API, enable the module, and then run the installation process.

If your site is in production mode, you will also need to run these commands to recompile sources:

```
php bin/magento setup:di:compile
php bin/magento setup:static-content:deploy
```

Step 3: Configure

See the configuration section below.

Updating Stripe

All extension updates are free. Just follow these directions to update to the latest version.

If you purchased from Magento Marketplace

Note, installing/upgrading this extension requires familiarity with your server's command line.

If you installed with composer, you can update using the following commands, in SSH at your site root:

```
composer update paradoxlabs/*
php bin/magento setup:upgrade
```

This will download and update to the latest extension version compatible with your system.

If your site is in production mode, you will also need to run these commands to recompile sources:

```
php bin/magento setup:di:compile
php bin/magento setup:static-content:deploy
```

If you purchased from store.paradoxlabs.com

Step 1: Upload files

Log into your account at store.paradoxlabs.com and download the latest version.

Open the extension archive and extract it onto your composer.

Upload all files within the **upload** folder into the root directory of Magento.

Folder in Download	Folder on Server
/upload/app/	→ /app/

Step 2: Run Update

If you are updating from version 4.x or older, first update the Stripe PHP library:

```
composer require "stripe/stripe-php:^20.0"
```

In SSH, from your site root, run the following commands:

```
php bin/magento setup:upgrade
```

If your site is in production mode, you will also need to run these commands to recompile sources:

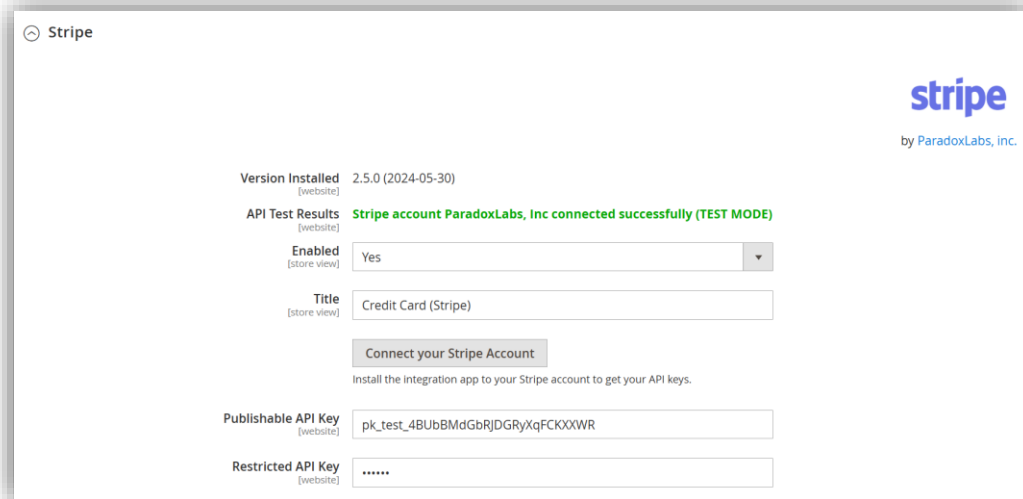
```
php bin/magento setup:di:compile
php bin/magento setup:static-content:deploy
```

Configuration

Before proceeding: Sign up for a [Stripe merchant account](#) if you don't have one already. You will need to activate your account before you can accept real payments.

Open your Admin Panel and go to **Admin > Stores > Settings > Configuration > Sales > Payment Methods**. Toward the bottom of the page, you'll find a 'Stripe' settings section like the below.

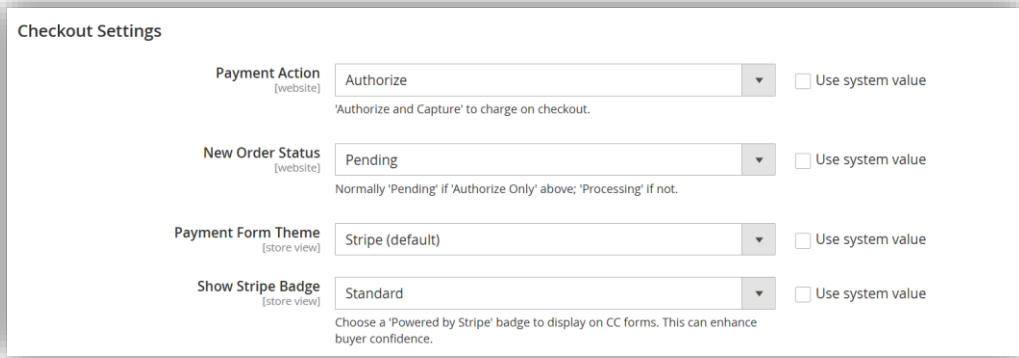
General



To get your API keys and connect the extension to your Stripe account, click the **“Connect your Stripe Account”** button. Install and authorize the ParadoxLabs app to your Stripe account, then copy the API keys from the app into the Magento settings.

- **Version Installed:** This tells you the version of our extension currently installed on your website. Please include this in any support requests.
- **API Test Results:** If you've entered your API Keys, we will automatically connect to Stripe to verify that the API works successfully. If we cannot connect to Stripe, or your info is incorrect, this will tell you with a red message. Correct the error, then reload the page and it should show **'Stripe account ... connected successfully.'** It will also show 'Test Mode' or 'Live Mode' based on the API keys you entered. Change your API keys to go from test mode to live mode, or vice versa.
- **Enable:** Yes to enable the payment method. If disabled, you will still be able to invoice/refund existing orders, but it will not show up as a payment option during checkout.
- **Title:** This controls the payment option label on checkout and order status pages.
- **Publishable API Key:** Get this from your Stripe account. Click 'Connect your Stripe Account', follow the prompts to install and authorize it, and then click "API Keys" to get your Publishable and Restricted API Keys for this extension.
- **Restricted API Key:** See above.
- **Enable Webhooks:** If yes, changes made in Stripe (captures, refunds, disputes, Radar reviews, stored card updates) are synced into Magento automatically. Saving the configuration with this enabled registers the webhook with Stripe. See the Webhooks section under Behavior Notes for details.

Checkout Settings



Checkout Settings

Payment Action [website] Authorize ☐ Use system value
'Authorize and Capture' to charge on checkout.

New Order Status [website] Pending ☐ Use system value
Normally 'Pending' if 'Authorize Only' above; 'Processing' if not.

Payment Form Theme [store view] Stripe (default) ☐ Use system value

Show Stripe Badge [store view] Standard ☐ Use system value
Choose a 'Powered by Stripe' badge to display on CC forms. This can enhance buyer confidence.

- **Payment Action:** Choose from the following options.
 - **Save info (do not authorize):** This will require customers to enter a credit card on checkout, and store that credit card in Stripe. The credit card will be validated in the process. No funds will be captured or held from the credit card upon checkout. Invoicing the order will perform a standalone authorize+capture transaction, but is not guaranteed to go through (funds may not be available).
 - **Authorize:** This will authorize the order amount upon checkout, allowing for manual invoicing and capture of the funds later. The order amount will be reserved for exactly 7 days. If you do not invoice within that time, the authorization will expire, and invoicing will perform a standalone authorize+capture transaction instead (which is not guaranteed to go through).
 - **Authorize and Capture:** This will capture all funds immediately when an order is placed. Payment processors strongly recommend not capturing funds unless/until you are within three days of fulfilling/shipping the order
- **New Order Status:** Set this to your desired initial status for orders paid via Stripe. Default Magento behavior is 'Pending' for Authorize Only, and 'Processing' for Authorize and Capture.
- **Payment Form Theme:** Select a theme for the payment form. Currently Stripe offers three options: Default, night (dark), and flat. Find more details in the [Stripe documentation](#).
- **Show Stripe Badge:** If yes, checkout will display a 'Powered by Stripe' badge next to the payment form.

Payment Settings

Payment Settings

Allow cards to not be stored
[website]

Yes

If yes, customers can choose whether to save their credit card during checkout.

Payment from Applicable Countries
[store view]

All Allowed Countries

Minimum Order Total
[store view]

Maximum Order Total
[store view]

Sort Order
[store view]

- Allow card to not be stored:** If yes, customers will have a 'Save for next time' checkbox on checkout. If no, logged in customers will see a message instead: *"For your convenience, this data will be stored securely by our payment processor."* Guests will never be given the option to store a credit card.
 Note that all cards are always stored in CIM, regardless of this setting or the customer's choice. This is necessary for payment processing. If the order is placed as a guest, or the customer chooses to not save their card, it will be automatically purged from all systems 120 days (the maximum refund period) after its last use. This ensures the info is available for edits, captures, and refunds, but respects the customer's wishes.
 If a card is 'not saved', it will not display under the customer's saved credit cards (Account > My Payment Data), nor will it be selectable during checkout. Note that as an admin, order 'edit' or 'reorder' will bypass this, always allowing reuse of the original payment info (unless it was since purged).
- Payment from Applicable Countries:** This setting allows you to limit which countries are able to use this payment method.
- Minimum Order Total:** This setting allows you to set a minimum order value for the payment option. For instance, set to 5 to only allow credit card checkout for orders of \$5 or more.
- Maximum Order Total:** This setting allows you to set a maximum order value for the payment option. For instance, set to 1000 to only allow credit card checkout for orders of \$1000 or less.
- Sort Order:** This setting allows you to change the order of payment options on checkout. Enter a number for this and all other payment methods according to the order you want them to display in.

Advanced Settings

Advanced Settings

Element Style

[store view]

```
{
  "base": {"fontSize": "16px"},
  "complete": {},
  "empty": {},
  "invalid": {}
}
```

Use this to change the font and color of the CC field on frontend checkout and card management. Must be valid JSON. See ['style' documentation for options](#).

Transaction Descriptor

[store view]

PDL MAGE2 %ORDERNO%

MAX 22 CHARS, letters/numbers/spaces only. This text will display on your customers' credit card statements. Use %ORDERNO% to include order number. Leave blank for none (default). [More info](#)

Reauthorize on Partial Invoice

[website]

Yes

If yes, when you create a partial invoice, we will reauthorize any outstanding balance on the order. This helps guarantee funds, but can cause multiple holds on the card until transactions settle.

Auto-select 'save for next time'

[website]

Yes

If yes, will be selected by default during checkout.

Verify SSL

[website]

Yes

Strongly recommended. Do not disable unless you get SSL errors and your host is unable to fix them.

- **Element style:** Use this setting to customize the font and colors of the frontend credit card inputs on checkout and account credit card management. The possible keys and values are described in Stripe documentation under 'variables': <https://docs.stripe.com/elements/appearance-api?platform=web#variables>
- **Transaction Descriptor:** Use this to customize the text that displays on your customers' credit card statements for each order. Use %ORDERNO% to include Magento's order number. Must be no more than 22 characters (letters, numbers, and spaces only)—any longer will be truncated. Leave it blank to use Stripe's default for your account. [More info on transaction descriptors](#)
- **Reauthorize on Partial Invoice:** If yes, and you invoice part of an order, a new authorization will be created for the outstanding order balance (if any). This helps guarantee funds. Any failure during reauthorization is ignored.
- **Auto-select 'save for next time':** If yes, the 'save this card for next time' checkbox will be checked by default. If no, customers will have to explicitly select it to store and reuse their card.
- **Verify SSL:** If yes, the Stripe API connection will be verified against known SSL information for the API. **Do not disable** unless you encounter SSL errors from the API test results and your host is unable to fix the underlying problem. Disabling this will make your store vulnerable to MITM (man-in-the-middle) attacks.
- **Cancel Incomplete Payments After (hours):** An hourly task will cancel abandoned/incomplete payment intents after this many hours, freeing up held authorizations for this store. Only affects payment intents with no associated order. Set to 0 to disable and use Stripe's default handling.

Behavior Notes

For the most part this is a standard Magento payment method, with the addition of the advanced Stored Card functionality and everything related to that. However, there are some things worth note:

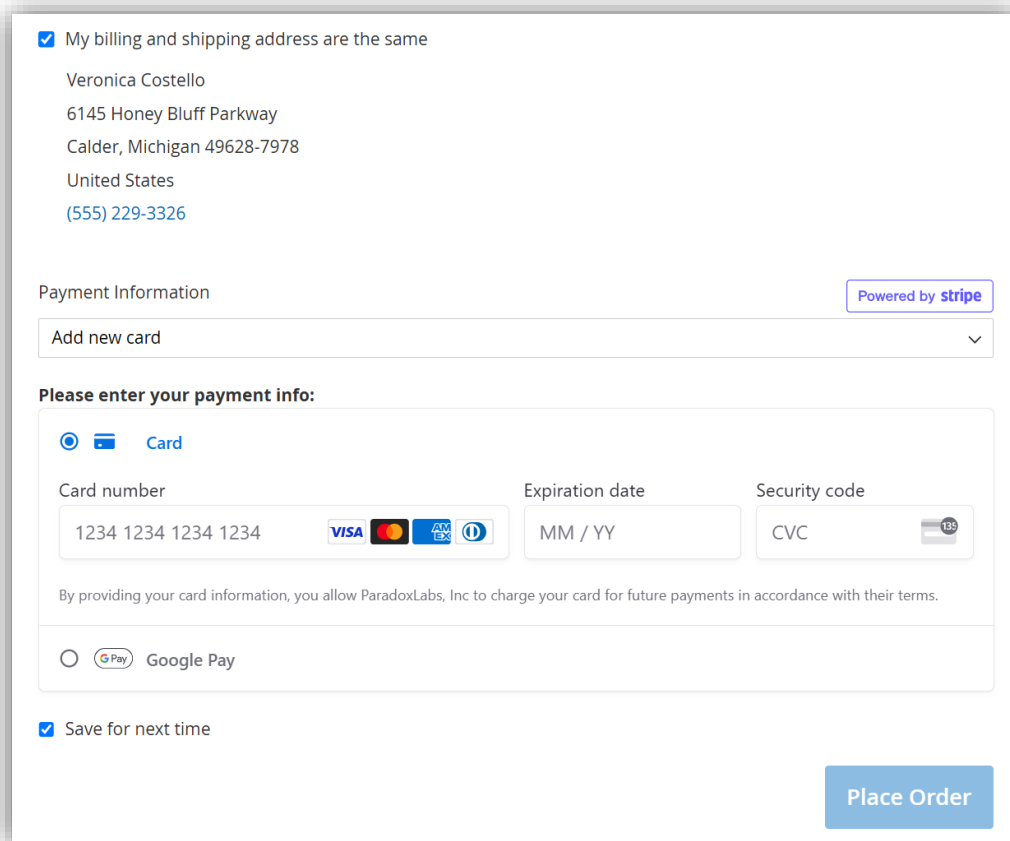
User Experience

One difference from standard Magento payment methods is that we use the unified Stripe Elements payment form everywhere, rather than inline credit card fields. Separators are auto-entered, type is auto-detected, and input moves seamlessly from each field to the next.

If you have multiple payment options enabled in your Stripe account, those options will be automatically presented to the customer when they are available (based on the user's region, browser, and other factors). This includes wallet payments like Apple Pay, Amazon Pay, and Google Pay; and possibly other options like ACH/bank account, Klarna, Affirm, Afterpay, and more.

Note: Apple Pay requires additional setup; you MUST verify each website domain in order to use Apply Pay.

[Follow Stripe's documentation to verify your domain.](#)



☒ My billing and shipping address are the same

Veronica Costello
6145 Honey Bluff Parkway
Calder, Michigan 49628-7978
United States
[\(555\) 229-3326](#)

Payment Information Powered by **stripe**

Add new card ▼

Please enter your payment info:

☒  Card

Card number 1234 1234 1234 1234  Expiration date MM / YY Security code CVC 

By providing your card information, you allow ParadoxLabs, Inc to charge your card for future payments in accordance with their terms.

☐  Google Pay

☒ Save for next time

Place Order

Security

Stripe Elements will always be active. There is no option to turn it off and revert to 'normal' credit card input. All credit card details (CC number, expiration date, CCV) are entered in an iframe hosted by Stripe, and transmitted

directly to Stripe's servers. At no time will credit card numbers ever touch your server for this payment method. This allows for the best possible mix of user experience and PCI compliance.

Some data about credit cards is kept in your database. This includes the card type, last 4 digits, expiration date, and identifiers, among other data. All of this data is available via the Stripe API, and no part of it is considered 'Confidential Cardholder Data' in the context of PCI compliance.

If you intend to collect payments from other sources using the Magento API, you will need to use the [Stripe API](#) to tokenize the credit card, then send that token to Magento.

Card Validation

Credit cards will always be validated before being stored.

Credit card validation codes will always be validated for new cards. For stored cards, the CCV was already validated, so it will not be required or requested again. The Stripe API does not provide any way to validate CCVs with stored-card transactions.

Card Storage

Credit cards will always be stored in Stripe to allow advanced functionality, even if the customer chooses not to save upon checkout. If the customer chooses not to save, or is a guest, the card will not show up for reuse within Magento. In this case, the card will be automatically purged from all systems after 120 days past its last use. This ensures the info is available for edits, captures, and refunds, but respects the customer's wishes.

There is no automatic card duplicate prevention: If a customer enters the same credit card as a new card 3 times, it will be stored within Stripe 3 times. There is no limit to the number of cards (sources) on a Stripe account.

Webhooks

New in version 5.0. When **Enable Webhooks** is turned on, the extension registers a webhook endpoint with Stripe and keeps Magento in sync with changes that happen outside of Magento:

- Captures and refunds performed in the Stripe Dashboard (or by any other system) automatically create the matching invoice or credit memo in Magento, along with payment transaction records.
- Disputes (chargebacks) and Radar fraud reviews place the affected order on hold, and release the hold when the dispute or review is resolved. Any capture or refund that arrives while the order is held is applied when the hold is released.
- Canceled or expired authorizations are noted on the order with a status comment. The order is not canceled automatically; invoicing afterward performs a standalone authorize+capture, the same as an expired authorization.
- Automatic card updates from the card networks refresh the stored card's details in Magento, and cards detached in Stripe are deactivated in Magento.

No setup is required in the Stripe Dashboard. The endpoint is registered automatically when you save the payment configuration with webhooks enabled. It sets up an endpoint per Stripe account, using your site's HTTPS base URL. A daily scheduled task reconciles the registration, and you can re-register at any time by saving the configuration or running `bin/magento paradoxlabs:stripe:register-webhooks`.

Webhooks require a Restricted API Key that includes webhook permission. If your key was created before this permission was added to the ParadoxLabs Stripe app, the configuration page will show a warning. If this happens,

update the ParadoxLabs app in your Stripe Dashboard to the latest version (at least 1.2.0), and copy the new Restricted API Key into the configuration to resolve it.

All incoming webhooks are verified by signature, and events are de-duplicated. Actions taken within Magento are recognized and will not be double-processed when Stripe echoes them back.

Usage

There isn't much to using this extension in practice: It's a standard Magento payment method, and all interfaces should be pretty self-explanatory. Here's a rundown of what you get:

Checkout Payment Form

The frontend payment form lets you choose/enter billing address and credit card. You can choose an existing card (if any) from the dropdown, or to add a new one.

The screenshot shows a checkout payment form with the following elements:

- ☒ My billing and shipping address are the same
- Billing Address:
 - Veronica Costello
 - 6145 Honey Bluff Parkway
 - Calder, Michigan 49628-7978
 - United States
 - (555) 229-3326
- Payment Information section:
 - Powered by stripe
 - Dropdown menu: Add new card
- Please enter your payment info:
 - ☒ Card
 - Card number: 1234 1234 1234 1234 (with VISA, Mastercard, Amex, and Discover logos)
 - Expiration date: MM / YY
 - Security code: CVC (with a 123 icon)
 - By providing your card information, you allow ParadoxLabs, Inc to charge your card for future payments in accordance with their terms.
 - ☐ Google Pay
- ☒ Save for next time
- Place Order button

Credit card type is detected automatically, and any format errors are immediately displayed below the input field.

Please enter your payment info:

☒  Card

Card number
4111 1111 1111 1110 

Expiration date
MM / YY

Security code
CVC 

Your card number is invalid.

By providing your card information, you allow ParadoxLabs, Inc to charge your card for future payments in accordance with their terms.

☐  Google Pay

If the customer has stored cards, their most recent one will be selected by default:

Select a new payment method:

☒ Credit Card (Stripe)

☒ My billing and shipping address are the same

John Doe
100 Main St.,
Lancaster, Pennsylvania 17063
United States
111-111-1111

Payment Information Powered by stripe

Visa XXXX-4242

[Apply Discount Code](#)

[Place Order](#)

Order Summary

Cart Subtotal	\$9.99
Shipping Free Shipping - Free	\$0.00
Order Total	\$9.99
1 Item in Cart	^



Medicine Ball (8lb)
Qty: 1

\$9.99

Ship To: 

John Doe
100 Main St.,
Lancaster, Pennsylvania 17063

Order status page

Order # 000000416 Pending

Order Date: May 18, 2017

[Reorder](#) [Print Order](#)

- [Account Dashboard](#)
- [Account Information](#)
- [Address Book](#)
- [My Orders](#)**
- [My Downloadable Products](#)
- [Newsletter Subscriptions](#)
- [Stored Payment Methods](#)
- [Billing Agreements](#)
- [My Product Reviews](#)
- [My Payment Data](#)
- [My Wish List](#)
- [My Subscriptions](#)

Recently Ordered

☒ Medicine Ball (8lb)

[Add to Cart](#) [View All](#)

[My Wish List](#)

Items Ordered

Items Ordered

Product Name	SKU	Price	Qty	Subtotal
Medicine Ball (8lb)	med-ball	\$9.99	Ordered: 1	\$9.99
Subtotal				\$9.99
Shipping & Handling				\$0.00
Grand Total				\$9.99

[Back to My Orders](#)

Order Information

Shipping Address

John Doe
100 Main St.
Lancaster, Pennsylvania, 17063
United States
T: 111-111-1111

Shipping Method

Free Shipping - Free

Billing Address

John Doe
100 Main St.
Lancaster, Pennsylvania, 17063
United States
T: 111-111-1111

Payment Method

Credit Card (Stripe)

Credit Card Type

Visa

Credit Card Number

XXXX-4242

Customer 'My Payment Options' account area

The My Payment Options section allows customers to see their stored cards, add, edit, and delete. When a customer deletes a stored card, they're prompted to confirm the removal.

My Account
My Orders
My Downloadable Products
My Wish List

Address Book
Account Information
Stored Payment Methods

My Product Reviews
Newsletter Subscriptions
My Payment Options
My Subscriptions

Compare Products
You have no items to compare.

Recently Ordered
☐ Sprite Foam Yoga Brick
[Add to Cart](#) [View All](#)

My Wish List
You have no items in your wish list.

My Payment Options

Credit Card (Stripe)

XXXX-4444 (Expires 05/2027)

Veronica Costello
6145 Honey Bluff Parkway
Calder, Michigan, 49628-7978
United States
T: (555) 229-3326
[Card In Use](#)

XXXX-3178 (Expires 08/2029)

Veronica Costello
6145 Honey Bluff Parkway
Calder, Michigan, 49628-7978
United States
T: (555) 229-3326
[Card In Use](#)

Add A Credit Card

Cardholder Information

First Name *

Last Name *

Street *

City *

State/Province *
Please select a region, state or province. ▼

Zip/Postal Code *

Country *
United States ▼

Telephone *

Payment Information

Payment Info *

☒ Card

Card number
1234 1234 1234 1234

Expiration date
MM / YY

Security code
CVC

By providing your card information, you allow ParadoxLabs, Inc to charge your card for future payments in accordance with their terms.

☐ Google Pay

[Save](#)

Note that cards associated with an open (uncaptured) order cannot be edited or deleted. They will display a 'Card In Use' message in place of the buttons. As soon as all orders paid by the card are completed, the 'Edit' and 'Delete' buttons will appear.

To prevent abuse, this section will only be available to customers after they have placed a successful order. If a customer attempts to access the page before then, they'll be redirected to the Account Dashboard with the message, "My Payment Options will be available after you've placed an order." Also to prevent abuse, if a customer receives errors trying to save a card five times, they will be blocked from access for 24 hours with the

message, "My Payment Options is currently unavailable. Please try again later." Both of these behaviors can be adjusted or disabled if necessary; please contact us if you have a problem.

Admin order form

The admin form displays the same options as frontend checkout.

Payment & Shipping Information

Payment Method

☒ Credit Card (Stripe)



Payment Info*

☒  Card

Card number



Expiration date

Security code



By providing your card information, you allow ParadoxLabs, Inc to charge your card for future payments in accordance with their terms.

☐  Google Pay

☒ Save for next time

Admin order status page

The admin panel shows extended payment info after placing an order, including transaction ID and validation results. These details are not visible to the customer at any time.

Address Information

Billing Address [Edit](#)

John Doe
100 Main St.
Lancaster, Pennsylvania, 17603
United States
T: 111-111-1111

Shipping Address [Edit](#)

John Doe
100 Main St.
Lancaster, Pennsylvania, 17063
United States
T: 111-111-1111

Payment & Shipping Method

Payment Information

Credit Card (Stripe)

Credit Card Type:	Visa
Credit Card Number:	XXXX-4242
Transaction ID:	ch_1AKs0SD52ltrz1NCaoz5xdUV
AVS Street-1 Check:	Pass
AVS Zipcode Check:	Pass
CVC Check:	Pass

The order was placed using USD.

Shipping & Handling Information

Free Shipping - Free \$0.00

Admin customer 'Payment Options' account area

Viewing a customer, you will see an added 'Payment Options' tab. This shows all of the same information with all of the same functionality as the equivalent frontend section. You can use this to view, add, edit, and delete your customers' stored Stripe payment data.

CUSTOMER INFORMATION

- Customer View
- Account Information
- Addresses
- Orders
- Billing Agreements
- Newsletter
- Product Reviews
- Payment Data**
- Wish List
- Subscriptions

Payment Data

Visa XXXX-4242 (Expires 01/2023) [Card In Use](#)

John Doe
100 Main St.
Lancaster, Pennsylvania, 17603
United States
T: 111-111-1111

Add A Credit Card

Cardholder Information

* First Name

* Last Name

* Street

Street Address 2

* City

* State/Province

* Zip/Postal Code

Admin transaction info

Viewing an order, you can also see full transaction info from the 'Transactions' tab.

#000000416

← Back Cancel Send Email Void Hold Invoice Ship Reorder Edit

ORDER VIEW

Search Reset Filter 1 records found 20 per page 1 of 1

ID	Order ID	Transaction ID	Parent Transaction ID	Payment Method	Transaction Type	Closed	Created
185	000000416	ch_1AKsUbDS2ltrz1NCPQ8Bbp2s		Credit Card (Stripe)	Authorization	No	May 18, 2017, 1:01:45 PM

Click into a transaction, and you'll see all of the raw transaction data returned by Stripe. This same data is accessible from the transaction record in code.

#ch_1AKsUbDS2ltrz1NCPQ8Bbp2s

Transaction Details

Key	Value
id	ch_1AKsUbDS2ltrz1NCPQ8Bbp2s
object	charge
amount	999
amount_refunded	0
application	
application_fee	
balance_transaction	
captured	0
created	1495128773
currency	usd
customer	cus_AgENBk5OAXhLwd
description	000000419
destination	
dispute	
failure_code	
failure_message	
invoice	
livemode	0
on_behalf_of	
order	
outcome.network_status	approved_by_network

Frequently Asked Questions & Troubleshooting

Please search our solution directory for the latest answers to common questions and issues:

<https://paradoxlabs.freshdesk.com/support/solutions>

If your question is not answered there, open a support ticket and we'll help you out.

Is ParadoxLabs Stripe PCI Compliance?

PCI compliance is a complex and multifaceted issue, covering every aspect of your business. We can't guarantee that your business is PCI-compliant. That depends on your server, passwords, business processes, regular security scans, any other payment methods, and a lot more. What we can tell you is that this extension will not prevent you from being PCI compliant. We don't log confidential cardholder data or do anything else that would bring you under scrutiny.

This extension implements **Stripe Elements** for all credit card forms, and does not support collecting credit card data by any other means. According to Stripe's official documentation, this makes the ParadoxLabs Stripe payment method eligible for **PCI v3.2 Self-Assessment Questionnaire A (SAQ A)**, the simplest possible SAQ form.

For more information, see the Stripe documentation: [PCI DSS guidelines](#)

All Stripe users must be compliant with the PCI Data Security Standards (PCI DSS). If you're using either [Stripe Checkout](#) or [Stripe Elements](#) to collect payment details, you are eligible for the simplest PCI validation method: [Self-Assessment Questionnaire A \(SAQ A\)](#). Elements and Checkout host all form inputs containing card data within an IFRAME served from Stripe's domain.

As long as you serve your payment pages over TLS, and use either Checkout or Elements as the only way of handling card information, Stripe automatically creates a combined SAQ A and Attestation of Compliance (AOC) for you. This pre-filled document is available in your account's [compliance settings](#).

Note that you **must** have SSL (TLS) enabled on all checkout and login forms, and that this eligibility **only** applies to this specific payment method. Any other payment methods or credit card handling your business may perform will have its own SAQ eligibility, and may require you to complete a more stringent SAQ form (A-EP or D).

For details on the SAQ types and what eligibility means, see "[Self-Assessment Questionnaire Instructions and Guidelines \(3.2\)](#)" (PDF, by PCI Standards Security Council).

For more information on Stripe's security policies and infrastructure, see Stripe documentation: [Security at Stripe](#)

Does ParadoxLabs Stripe support SCA?

As of extension version 3.0, yes, we fully support SCA and 3D Secure 2.0.

How do I do an online refund from Magento?

In order to process an 'online' refund through Stripe, you have to go to the **invoice** you want to refund, and click the 'Credit Memo' button from there.

If you've done that correctly, at the bottom of the page you should see a button that says 'Refund'.

As of version 3.2, clicking 'Credit Memo' from the order will automatically load the most recent invoice, so the online 'Refund' option is available. If you still see only a 'Refund Offline' button, no invoice was loaded — open the specific invoice you want to refund and click 'Credit Memo' from there.

The reason for this is that the refund needs to be associated with a particular capture transaction. An order can contain any number of capture transactions, but every capture has an invoice that's directly related. You refund an invoice, not an order.

Do you support Apple Pay?

As of version 2.3.0, yes! If you're on an older version, update to the latest release, then set up Apple Pay in your Stripe account. You must verify your domain with Stripe to use Apple Pay.

How does this payment method handle currency?

Transactions are automatically processed in the base currency for the website customers are purchasing from. Any alternate currencies selected on the frontend are converted to the website's base currency by Magento's built-in currency handling, based on conversion rates provided by a web service configured in your Magento Admin Panel.

Magento allows for a separate base currency per website, if configured to do so. In order to define explicit prices in multiple currencies, each currency must have its own website where it is set as the base currency. All currency setup is configured outside of our payment extension settings.

Technical / Integration Details

Architecture

The payment method code for this method is `paradoxlabs_stripe`.

`ParadoxLabs_Stripe` is the payment method module, built heavily on the `ParadoxLabs-TokenBase` module. `TokenBase` defines a variety of interfaces and architecture for handling tokenization and stored cards cleanly.

The payment method class is `\ParadoxLabs\Stripe\Model\Method`. This talks to Stripe through `\ParadoxLabs\Stripe\Model\Gateway` (via proxy methods for the Stripe PHP library), and stores card metadata in instances of `\ParadoxLabs\Stripe\Model\Card`. Each of these extends an equivalent abstract class in `TokenBase`, and implements only the details specific to the Stripe API.

Card instances are stored in table `paradoxlabs_stored_card`, and referenced by quotes and orders via a `tokenbase_id` column on tables `quote_payment` and `sales_order_payment`.

In all cases, we strongly discourage any customization by editing our code directly. We cannot support customizations. Use Magento's preferences or plugins to modify behavior if necessary. If your use case isn't covered, let us know.

Custom customer attributes

- `stripe_customer_id`

Custom database schema

- Added table: `paradoxlabs_stored_card`
- Added table: `paradoxlabs_stripe_webhook_event`
- Added column: `quote_payment.tokenbase_id`
- Added column: `sales_order_payment.tokenbase_id`

Events

- `tokenbase_before_load_payment_info` (method, customer, transport, info): Fires before preparing method-specific information for the order payment info blocks (frontend, admin, and emails). Use this to add additional information to the payment info block.
- `tokenbase_after_load_payment_info` (method, customer, transport, info): Fires before preparing method-specific information for the order payment info blocks (frontend, admin, and emails). Use this to add additional information to the payment info block, or modify what's there by default.
- `tokenbase_before_load_active_cards` (method, customer): Fires before loading a customer's available stored cards.
- `tokenbase_after_load_active_cards` (method, customer, cards): Fires after loading a customer's available stored cards. Use this to modify cards available to the customer or admin.

Magento API: REST and SOAP

This module supports the Magento API via standard interfaces. You can use it to create, read, update, and delete stored cards.

If you have a specific use case in mind that is not covered, please let us know.

You can generate new cards by creating an order with our payment method (code `paradoxlabs_stripe`), and a valid Stripe ConfirmationToken (`ctoken_...`) as `additional_data.token`. To place an order with a stored card, pass that card's hash in as `additional_data.card_id`.

Note that raw credit card numbers (`cc_number`) will not be accepted. You must collect the payment details with Stripe.js and pass in the resulting ConfirmationToken (`ctoken_...`). See Stripe documentation on ConfirmationTokens for details.

Strong Customer Authentication (SCA / 3D Secure): If the customer's bank requires authentication, order placement will fail with the message "Additional authentication is required to complete your payment. Please follow the prompts to verify your card and try again." The error's parameters include `intent_id` and `client_secret` for the pending PaymentIntent. Complete the authentication client-side with Stripe.js (`stripe.handleNextAction` using the `client_secret`), then set the payment method again passing the PaymentIntent id as the token and place the order again.

Available REST API requests below. Some response data has been omitted for brevity.

Create and update (POST, PUT) requests take three objects: `card` with primary card data, `address` with address information, and `additional` for card metadata. In responses, `address` and `additional` will be nested within `card` as `address_object` and `additional_object`. This is done for technical reasons. The data formats differ, and not all fields that are returned can be set via API (EG `in_use`, `label`). This means you cannot take a card record and directly post it back to the API to update.

Integration / Admin-Authenticated API Endpoints

These API requests allow solutions acting with an admin user login, OAUTH authentication, or token-based authentication to take action on any card in the system. Data and behavior are not limited.

GET /V1/tokenbase/:cardId (get one card by ID)

Example request:

```
GET /rest/v1/tokenbase/1 HTTP/1.1
Host: {host}
Authorization: Bearer {api_key}
```

Example response:

```
{
  "id": 1,
  "in_use": true,
  "additional_object": {
    "cc_type": "DI",
    "cc_last4": "0012",
    "cc_exp_year": "2019",
    "cc_exp_month": "6",
    "cc_country": "US",
    "object": "card",
    "funding": "credit",
    "address_line1_check": "pass",
    "cvc_check": "pass",
```

```
{
  "fingerprint": "0i4PDLs7u409euEo"
},
"address_object": {
  "region": {
    "region_code": "PA",
    "region": "Pennsylvania",
    "region_id": 51
  },
  "region_id": 51,
  "country_id": "US",
  "street": [
    "123 Test Ln."
  ],
  "company": "",
  "telephone": "111-111-1111",
  "postcode": "17603",
  "city": "Lancaster",
  "firstname": "John",
  "lastname": "Doe"
},
"customer_email": "email@example.com",
"customer_id": 1,
"customer_ip": "127.0.0.1",
"profile_id": "cus_Abcdeg3n490ds8",
"payment_id": "card_1ABcPDLS7udh38sdoivnw98Q",
"method": "paradoxlabs_stripe",
"hash": "f7d085165acdfa0ea6a0b...770111",
"active": "1",
"created_at": "2017-08-03 16:31:54",
"updated_at": "2017-09-20 14:24:14",
"last_use": "2017-08-03 16:31:54",
"expires": "2019-06-30 23:59:59",
"label": "Discover xxxx-0012"
}
```

GET /V1/tokenbase/search (get multiple cards, with searchCriteria)

Example request:

```
GET /rest/V1/tokenbase/search?searchCriteria[pageSize]=1 HTTP/1.1
Host: {host}
Authorization: Bearer {api_key}
```

Example response:

```
{
  "items": [
    {
      "id": 1,
      // ... other card info
    }
  ],
  "search_criteria": {
    "filter_groups": [],
    "page_size": 1
  },
  "total_count": 51
}
```

See also: [Search using REST APIs](#) (Magento DevDocs)

POST /V1/tokenbase (create card)

Example request:

```
POST /rest/V1/tokenbase HTTP/1.1
Host: {host}
Authorization: Bearer {api_key}
Content-Type: application/json

{
  "card": {
    "customer_email": "email@example.com",
    "customer_id": 1,
    "customer_ip": "",
```

```

    "profile_id": "cus_Abcdeg3n490ds8",
    "payment_id": "card_1ABcPDLs7udh38sdoivnw98Q",
    "method": "paradoxlabs_stripe",
    "active": "1",
    "created_at": "2017-08-03 16:31:54",
    "last_use": "2017-08-03 16:31:54",
    "expires": "2019-06-30 23:59:59"
  },
  "address": {
    "region": {
      "region_code": "PA",
      "region": "Pennsylvania",
      "region_id": 51
    },
    "region_id": 51,
    "country_id": "US",
    "street": [
      "123 Test Ln."
    ],
    "company": "",
    "telephone": "111-111-1111",
    "postcode": "17603",
    "city": "Lancaster",
    "firstname": "John",
    "lastname": "Doe",
    "vat_id": ""
  },
  "additional": {
    "cc_exp_month": "06",
    "cc_exp_year": "2019",
    "cc_last4": "0012",
    "cc_type": "DI",
    "cc_country": "US",
    "token": "ctoken_12345...",
    "object": "card",
    "funding": "credit",
    "address_line1_check": "pass",
    "cvc_check": "pass",
    "fingerprint": "0i4PDLs7u409euEo"
  }
}

```

Example response:

```

{
  "id": 95,
  "in_use": false,
  "additional_object": {
    "cc_type": "DI",
    "cc_last4": "0012",
    "cc_exp_year": "2019",
    "cc_exp_month": "06",
    "cc_country": "US",
    "object": "card",
    "funding": "credit",
    "address_line1_check": "pass",
    "cvc_check": "pass",
    "fingerprint": "0i4PDLs7u409euEo"
  },
  "address_object": {
    "region": {
      "region_code": "PA",
      "region": "Pennsylvania",
      "region_id": 51
    },
    "region_id": 51,
    "country_id": "US",
    "street": [
      "123 Test Ln."
    ],
    "company": "",
    "telephone": "111-111-1111",
    "postcode": "17603",
    "city": "Lancaster",
    "firstname": "John",
    "lastname": "Doe",
  },
  "customer_email": "email@example.com",
}

```



```

"customer_id": 1,
"customer_ip": "127.0.0.1",
"profile_id": "cus_Abcdeg3n490ds8",
"payment_id": "card_1ABcPDls7udh38sdoivnw98Q",
"method": "paradoxlabs_stripe",
"hash": "9b83d4683f3d3...2309ccd65b",
"active": "1",
"created_at": "2017-09-25 17:41:21",
"updated_at": "2017-09-25 17:41:21",
"last_use": "2017-08-03 16:31:54",
"expires": "2019-06-30 23:59:59",
"label": "Discover XXXX-0012"
}

```

PUT /V1/tokenbase/:cardId (update card)

Example request:

```

PUT /rest/v1/tokenbase/1 HTTP/1.1
Host: {host}
Authorization: Bearer {api_key}
Content-Type: application/json

```

```

{
  "card": {
    "id": 1,
    "customer_email": "email@example.com",
    "customer_id": 1,
    "customer_ip": "127.0.0.1",
    "profile_id": "cus_Abcdeg3n490ds8",
    "payment_id": "card_1ABcPDls7udh38sdoivnw98Q",
    "method": "paradoxlabs_stripe",
    "hash": "f7d085165acdfa0ea6a0b...770111",
    "active": "1",
    "created_at": "2017-08-03 16:31:54",
    "last_use": "2017-08-03 16:31:54",
    "expires": "2019-06-30 23:59:59"
  },
  "address": {
    "region": {
      "region_code": "PA",
      "region": "Pennsylvania",
      "region_id": 51
    },
    "region_id": 51,
    "country_id": "US",
    "street": [
      "123 Test Ln."
    ],
    "company": "",
    "telephone": "111-111-1111",
    "postcode": "17603",
    "city": "Lancaster",
    "firstname": "John",
    "lastname": "Doe",
    "vat_id": ""
  },
  "additional": {
    "cc_exp_month": "06",
    "cc_exp_year": "2019",
    "cc_last4": "0012",
    "cc_type": "DI",
    "cc_country": "US",
    "token": "ctoken_12345...",
    "object": "card",
    "funding": "credit",
    "address_line1_check": "pass",
    "cvc_check": "pass",
    "fingerprint": "0i4PDls7u409euEo"
  }
}

```

Example response:

```
{
  "id": 1,
  "in_use": false,
  "additional_object": {
    "cc_type": "DI",
    "cc_last4": "0012",
    "cc_exp_year": "2019",
    "cc_exp_month": "06",
    "cc_country": "US",
    "object": "card",
    "funding": "credit",
    "address_line1_check": "pass",
    "cvc_check": "pass",
    "fingerprint": "0i4PDLs7u409euEo"
  },
  "address_object": {
    "region": {
      "region_code": "PA",
      "region": "Pennsylvania",
      "region_id": 51
    },
    "region_id": 51,
    "country_id": "US",
    "street": [
      "123 Test Ln."
    ],
    "company": "",
    "telephone": "111-111-1111",
    "postcode": "17603",
    "city": "Lancaster",
    "firstname": "John",
    "lastname": "Doe",
  },
  "customer_email": "email@example.com",
  "customer_id": 1,
  "customer_ip": "127.0.0.1",
  "profile_id": "cus_Abcdeg3n490ds8",
  "payment_id": "card_1ABcPDLs7udh38sdoivnw98Q",
  "method": "paradoxlabs_stripe",
  "hash": "f7d085165acdfa0ea6a0b...770111",
  "active": "1",
  "created_at": "2017-09-25 17:41:21",
  "updated_at": "2017-09-25 17:41:21",
  "last_use": "2017-08-03 16:31:54",
  "expires": "2019-06-30 23:59:59",
  "label": "Discover xxxx-0012"
}
```

DELETE /V1/tokenbase/:cardId (delete card by ID)

Example request:

```
DELETE /rest/V1/tokenbase/95 HTTP/1.1
Host: {host}
Authorization: Bearer {api_key}
```

Example response:

```
True
```

Customer Authenticated API Endpoints

These API requests allow authenticated frontend customers to manage their stored cards. This is intended for headless implementations or app integration where card management needs to be exposed outside of Magento's standard frontend.

Customers will only be able to access and manipulate active cards assigned to their specific customer ID.

Note: These requests are disabled by default. You can enable them at **Admin > Stores > Configuration > Sales > Checkout > ParadoxLabs Payment Module Settings > Enable public API**. Only enable this if you use them.

GET /V1/tokenbase/mine/:cardHash (get one card by hash)

Example request:

```
GET /rest/v1/tokenbase/mine/50b8e326b012e793957215c0361afc4b52434b26 HTTP/1.1
Host: {host}
Authorization: Bearer {api_key}
```

Example response:

```
{
  "id": 1,
  "in_use": true,
  "additional_object": {
    "cc_type": "DI",
    "cc_last4": "0012",
    "cc_exp_year": "2019",
    "cc_exp_month": "6"
  },
  "address_object": {
    "region": {
      "region_code": "PA",
      "region": "Pennsylvania",
      "region_id": 51
    },
    "region_id": 51,
    "country_id": "US",
    "street": [
      "123 Test Ln."
    ],
    "company": "",
    "telephone": "111-111-1111",
    "postcode": "17603",
    "city": "Lancaster",
    "firstname": "John",
    "lastname": "Doe"
  },
  "customer_email": "email@example.com",
  "customer_id": 1,
  "customer_ip": "127.0.0.1",
  "profile_id": "1234567890",
  "payment_id": "0987654321",
  "method": "paradoxlabs_stripe",
  "hash": "50b8e326b012e793957215c0361afc4b52434b26",
  "active": "1",
  "created_at": "2017-08-03 16:31:54",
  "updated_at": "2017-09-20 14:24:14",
  "last_use": "2017-08-03 16:31:54",
  "expires": "2019-06-30 23:59:59",
  "label": "Discover XXXX-0012"
}
```

GET /V1/tokenbase/mine/search (get multiple cards, with searchCriteria)

Example request:

```
GET /rest/v1/tokenbase/mine/search?searchCriteria[pageSize]=3 HTTP/1.1
Host: {host}
Authorization: Bearer {api_key}
```

Example response:

```
{
  "items": [
    {
      "id": 1,
      // ... other card info
    }
  ],
  "search_criteria": {
    "filter_groups": [],
    "page_size": 3
  },
  "total_count": 5
}
```

```
}
```

See also: [Search using REST APIs](#) (Magento DevDocs)

POST /V1/tokenbase/mine (create card)

Example request:

```
POST /rest/V1/tokenbase/mine HTTP/1.1
Host: {host}
Content-Type: application/json
Authorization: Bearer {api_key}

{
  "card": {
    "customer_email": "email@example.com",
    "customer_id": 1,
    "customer_ip": "127.0.0.1",
    "profile_id": "1234567890",
    "payment_id": "0987654321",
    "method": "paradoxlabs_stripe",
    "active": "1"
  },
  "address": {
    "region": {
      "region_code": "PA",
      "region": "Pennsylvania",
      "region_id": 51
    },
    "region_id": 51,
    "country_id": "US",
    "street": [
      "123 Test Ln."
    ],
    "company": "",
    "telephone": "111-111-1111",
    "postcode": "17603",
    "city": "Lancaster",
    "firstname": "John",
    "lastname": "Doe",
    "vat_id": ""
  },
  "additional": {
    "cc_exp_month": "06",
    "cc_exp_year": "2019",
    "cc_last4": "0012",
    "cc_type": "DI",
    "token": "ctoken_12345..."
  }
}
```

Example response:

```
{
  "id": 95,
  "in_use": false,
  "additional_object": {
    "cc_type": "DI",
    "cc_last4": "0012",
    "cc_exp_year": "2019",
    "cc_exp_month": "06"
  },
  "address_object": {
    "region": {
      "region_code": "PA",
      "region": "Pennsylvania",
      "region_id": 51
    },
    "region_id": 51,
    "country_id": "US",
    "street": [
      "123 Test Ln."
    ],
    "company": "",
    "telephone": "111-111-1111",
    "postcode": "17603",

```

```

    "city": "Lancaster",
    "firstname": "John",
    "lastname": "Doe",
  },
  "customer_email": "email@example.com",
  "customer_id": 1,
  "customer_ip": "127.0.0.1",
  "profile_id": "1234567890",
  "payment_id": "0987654321",
  "method": "paradoxlabs_stripe",
  "hash": "9b83d4683f3d3...2309ccd65b",
  "active": "1",
  "created_at": "2017-09-25 17:41:21",
  "updated_at": "2017-09-25 17:41:21",
  "last_use": "2017-08-03 16:31:54",
  "expires": "2019-06-30 23:59:59",
  "label": "Discover xxxx-0012"
}

```

PUT /V1/tokenbase/mine/:cardHash (update card by hash)

Example request:

```

PUT /rest/v1/tokenbase/mine/50b8e326b012e793957215c0361afc4b52434b26 HTTP/1.1
Host: {host}
Authorization: Bearer {api_key}
Content-Type: application/json

```

```

{
  "card": {
    "id": 1,
    "customer_email": "email@example.com",
    "customer_id": 1,
    "customer_ip": "127.0.0.1",
    "profile_id": "1234567890",
    "payment_id": "0987654321",
    "method": "paradoxlabs_stripe",
    "hash": "50b8e326b012e793957215c0361afc4b52434b26",
    "active": "1",
    "expires": "2019-06-30 23:59:59"
  },
  "address": {
    "region": {
      "region_code": "PA",
      "region": "Pennsylvania",
      "region_id": 51
    },
    "region_id": 51,
    "country_id": "US",
    "street": [
      "123 Test Ln."
    ],
    "company": "",
    "telephone": "111-111-1111",
    "postcode": "17603",
    "city": "Lancaster",
    "firstname": "John",
    "lastname": "Doe",
    "vat_id": ""
  },
  "additional": {
    "cc_exp_month": "06",
    "cc_exp_year": "2019",
    "cc_last4": "0012",
    "cc_type": "DI"
  }
}

```

Example response:

```

{
  "id": 1,
  "in_use": false,
  "additional_object": {
    "cc_type": "DI",
    "cc_last4": "0012",
    "cc_exp_year": "2019",

```

```
{
  "cc_exp_month": "06"
},
"address_object": {
  "region": {
    "region_code": "PA",
    "region": "Pennsylvania",
    "region_id": 51
  },
  "region_id": 51,
  "country_id": "US",
  "street": [
    "123 Test Ln."
  ],
  "company": "",
  "telephone": "111-111-1111",
  "postcode": "17603",
  "city": "Lancaster",
  "firstname": "John",
  "lastname": "Doe",
},
"customer_email": "email@example.com",
"customer_id": 1,
"customer_ip": "127.0.0.1",
"profile_id": "1234567890",
"payment_id": "0987654321",
"method": "paradoxlabs_stripe",
"hash": "50b8e326b012e793957215c0361afc4b52434b26",
"active": "1",
"created_at": "2017-09-25 17:41:21",
"updated_at": "2017-09-25 17:41:21",
"last_use": "2017-08-03 16:31:54",
"expires": "2019-06-30 23:59:59",
"label": "Discover xxxx-0012"
}
```

DELETE /V1/tokenbase/mine/:cardHash (delete card by hash)

Example request:

```
DELETE /rest/v1/tokenbase/mine/50b8e326b012e793957215c0361afc4b52434b26 HTTP/1.1
Host: {host}
Authorization: Bearer {api_key}
```

Example response:

```
True
```

Guest API Endpoints

These API requests allow unauthenticated frontend guest users to add and fetch an individual stored card. This is intended for headless implementations or app integration where card management needs to be exposed outside of Magento's standard frontend. Guests are not able to list, edit, delete, or reuse stored cards, so no API requests are exposed for those actions.

Guests will only be able to access and manipulate active cards, by hash, not assigned to any customer ID.

Note: These requests are disabled by default. You can enable them at **Admin > Stores > Configuration > Sales > Checkout > ParadoxLabs Payment Module Settings > Enable public API**. Only enable this if you use them.

GET /V1/tokenbase/guest/:cardHash (get one card by hash)

Example request:

```
GET /rest/v1/tokenbase/guest/50b8e326b012e793957215c0361afc4b52434b26 HTTP/1.1
Host: {host}
```

Example response:

```
{
  "id": 1,
  "in_use": true,
  "additional_object": {
    "cc_type": "DI",
    "cc_last4": "0012",
    "cc_exp_year": "2019",
    "cc_exp_month": "6"
  },
  "address_object": {
    "region": {
      "region_code": "PA",
      "region": "Pennsylvania",
      "region_id": 51
    },
    "region_id": 51,
    "country_id": "US",
    "street": [
      "123 Test Ln."
    ],
    "company": "",
    "telephone": "111-111-1111",
    "postcode": "17603",
    "city": "Lancaster",
    "firstname": "John",
    "lastname": "Doe"
  },
  "customer_email": "email@example.com",
  "customer_id": 0,
  "customer_ip": "127.0.0.1",
  "profile_id": "1234567890",
  "payment_id": "0987654321",
  "method": "paradoxlabs_stripe",
  "hash": "50b8e326b012e793957215c0361afc4b52434b26",
  "active": "1",
  "created_at": "2017-08-03 16:31:54",
  "updated_at": "2017-09-20 14:24:14",
  "last_use": "2017-08-03 16:31:54",
  "expires": "2019-06-30 23:59:59",
  "label": "Discover xxxx-0012"
}
```

POST /V1/tokenbase/guest (create card)

Example request:

```
POST /rest/V1/tokenbase/guest HTTP/1.1
Host: {host}
Content-Type: application/json

{
  "card": {
    "customer_email": "email@example.com",
    "customer_id": 0,
    "customer_ip": "127.0.0.1",
    "profile_id": "1234567890",
    "payment_id": "0987654321",
    "method": "paradoxlabs_stripe",
    "active": "1"
  },
  "address": {
    "region": {
      "region_code": "PA",
      "region": "Pennsylvania",
      "region_id": 51
    },
    "region_id": 51,
    "country_id": "US",
    "street": [
      "123 Test Ln."
    ],
    "company": "",
    "telephone": "111-111-1111",
    "postcode": "17603",
    "city": "Lancaster",
    "firstname": "John",
    "lastname": "Doe",
    "vat_id": ""
  }
}
```

```

    "additional": {
      "cc_exp_month": "06",
      "cc_exp_year": "2019",
      "cc_last4": "0012",
      "cc_type": "DI",
      "token": "ctoken_12345..."
    }
  }
}

```

Example response:

```

{
  "id": 95,
  "in_use": false,
  "additional_object": {
    "cc_type": "DI",
    "cc_last4": "0012",
    "cc_exp_year": "2019",
    "cc_exp_month": "06"
  },
  "address_object": {
    "region": {
      "region_code": "PA",
      "region": "Pennsylvania",
      "region_id": 51
    },
    "region_id": 51,
    "country_id": "US",
    "street": [
      "123 Test Ln."
    ],
    "company": "",
    "telephone": "111-111-1111",
    "postcode": "17603",
    "city": "Lancaster",
    "firstname": "John",
    "lastname": "Doe",
  },
  "customer_email": "email@example.com",
  "customer_id": 0,
  "customer_ip": "127.0.0.1",
  "profile_id": "1234567890",
  "payment_id": "0987654321",
  "method": "paradoxlabs_stripe",
  "hash": "9b83d4683f3d3...2309ccd65b",
  "active": "1",
  "created_at": "2017-09-25 17:41:21",
  "updated_at": "2017-09-25 17:41:21",
  "last_use": "2017-08-03 16:31:54",
  "expires": "2019-06-30 23:59:59",
  "label": "Discover xxxx-0012"
}

```

Magento API: GraphQL

For Magento 2.3.1+, this extension supports the GraphQL API for all customer card management. This is intended for PWA and headless implementations where card management needs to be exposed outside of Magento's standard frontend.

Customers will only be able to access and manipulate active cards assigned to their specific customer ID.

Guests will only be able to access and manipulate active cards, by hash, not assigned to any customer ID.

Note: These requests are disabled by default. You can enable them at **Admin > Stores > Configuration > Sales > Checkout > ParadoxLabs Payment Module Settings > Enable public API**. Only enable this if you use them.

We recommend using [GraphiQL](#) or the Chrome [ChromeiQL browser extension](#) for browsing your store's GraphQL schema and testing API requests.

Note that raw credit card numbers (cc_number) will not be accepted. You must collect the payment details with Stripe.js and pass in the resulting ConfirmationToken (ctoken_...) as token. See Stripe documentation on ConfirmationTokens for details.

Strong Customer Authentication (SCA / 3D Secure): If the customer's bank requires authentication, order placement will fail with the message "Additional authentication is required to complete your payment. Please follow the prompts to verify your card and try again." The error's parameters include intent_id and client_secret for the pending PaymentIntent. Complete the authentication client-side with Stripe.js (stripe.handleNextAction using the client_secret), then set the payment method again passing the PaymentIntent id as the token and place the order again.

Queries

tokenBaseCards(hash: String): [TokenBaseCard]

Get the current customer's stored card(s), if any. Takes a card hash (optional); returns one or more TokenBaseCard records. If no hash is given, will return all active cards belonging to the customer.

tokenBaseCheckoutConfig(method: String!): TokenBaseCheckoutConfig

Get checkout configuration for the given TokenBase payment method. Takes a TokenBase payment method code, such as paradoxlabs_stripe; returns a TokenBaseCheckoutConfig. This returns all data necessary to render and handle the client-side checkout form. Values mirror what is passed to Magento's standard frontend checkout.

Mutations

createTokenBaseCard(input: TokenBaseCardCreateInput!): TokenBaseCard

Create a new stored card. Takes TokenBaseCardCreateInput, returns the new stored TokenBaseCard if successful.

deleteTokenBaseCard(hash: String!): Boolean

Delete a stored card. Takes a card hash; returns true if successful.

updateTokenBaseCard(input: TokenBaseCardUpdateInput!): TokenBaseCard

Update an existing stored card. Takes TokenBaseCardUpdateInput; returns the updated TokenBaseCard if successful.

Data Types

TokenBaseCard

A stored payment account/credit card.

```
type TokenBaseCard {
  hash: String           Card identifier hash
  address: CustomerAddress Card billing address
  customer_email: String Customer email
  customer_id: Int       Customer ID
  customer_ip: String     Created-by IP
  profile_id: String      Card gateway profile ID
  payment_id: String      Card gateway payment ID
  method: String          Payment method code
  active: Boolean         Is card active
  created_at: String       Created-at date
  updated_at: String       Last updated date
  last_use: String         Last used date
  expires: String          Expiration date
  label: String            Card label
  additional: TokenBaseCardAdditional Card payment data
}
```

TokenBaseCardAdditional

Details and metadata for a stored CC/ACH.

type TokenBaseCardAdditional {	
cc_type: String	CC Type
cc_owner: String	CC Owner
cc_bin: String	CC Bin (CC First-6)
cc_last4: String	CC Last-4
cc_exp_year: String	CC Expiration Year
cc_exp_month: String	CC Expiration Month
echeck_account_name: String	ACH Account Name
echeck_bank_name: String	ACH Bank Name
echeck_account_type: TokenBaseEcheckAccountType	ACH Account type
echeck_routing_number_last4: String	ACH Routing Number Last-4
echeck_account_number_last4: String	ACH Account Number Last-4
}	

TokenBaseCheckoutConfig

Checkout configuration for a TokenBase payment method.

type TokenBaseCheckoutConfig {	
method: String	Payment method code
useVault: Boolean	Are stored cards enabled?
canSaveCard: Boolean	Can cards be saved?
forceSaveCard: Boolean	Is card saving forced?
defaultSaveCard: Boolean	Hash of the default card to select
isCCDetectionEnabled: Boolean	Is CC type detection enabled?
logoImage: String	Payment logo image URL (if enabled)
requireCcv: Boolean	Is CVV required for stored cards?
sandbox: Boolean	Is the payment gateway in sandbox mode?
canStoreBin: Boolean	Is CC BIN (CC first6) storage enabled?
availableTypes: [TokenBaseKeyValue]	Available CC types
months: [TokenBaseKeyValue]	Available CC Exp Months
years: [TokenBaseKeyValue]	Available CC Exp Years
hasVerification: Boolean	Is CVV enabled?
cvvImageUrl: String	CVV helper image URL
publicKey: String	Stripe API public key
country: String	Transaction country
currency: String	Transaction currency
grandTotal: Float	Transaction grand total
mode: String	Stripe intent scope: 'payment' or 'setup'
elementsTheme: String	Configured payment form theme
}	

TokenBaseKeyValue

Container for generic key/value data.

type TokenBaseKeyValue {	
key: String	Generic key
value: String	Generic value
}	

TokenBaseCardUpdateInput

Input for updating a stored card.

input TokenBaseCardUpdateInput {	
hash: String!	Card identifier hash to update (required)
address: CustomerAddressInput	Card billing address
customer_email: String	Customer email
customer_ip: String	Created-by IP
method: String	Payment method code
active: Boolean	Is card active
expires: String	Card expiration date (YYYY-MM-DD 23:59:59)
additional: TokenBaseCardPaymentInput	Card payment data
}	

TokenBaseCardCreateInput

Input for creating a stored card.

input TokenBaseCardCreateInput {	
address: CustomerAddressInput	Card billing address
customer_email: String!	Customer email (required)
customer_ip: String	Created-by IP

method: String!	Payment method code (required)
active: Boolean	Is card active
expires: String	Card expiration date (YYYY-MM-DD 23:59:59)
additional: TokenBaseCardPaymentInput	Card payment data

TokenBaseCardPaymentInput

Payment data for a stored card. Note, the specific fields that are relevant depend on the payment method. This data structure is also used for adding payment data to the cart during checkout.

input TokenBaseCardPaymentInput {	
cc_type: String	CC Type
cc_owner: String	CC Owner
cc_bin: String	CC Bin (CC First-6)
cc_last4: String	CC Last-4
cc_number: String	CC Number
cc_cid: String	CC CVV
cc_exp_year: String	CC Expiration Year
cc_exp_month: String	CC Expiration Month
echeck_account_name: String	ACH Account Name
echeck_bank_name: String	ACH Bank Name
echeck_account_type: TokenBaseEcheckAccountType	ACH Account Type
echeck_routing_number: String	ACH Routing Number
echeck_account_number: String	ACH Account Number
token: String	Stripe ConfirmationToken (ctoken_...)
save: Boolean	Save the card for later use? (checkout only)
card_id: String	Card identifier hash to use (checkout only)
}	

GraphQL Query Examples

Some response data has been omitted for brevity.

Fetch card by ID

Example request:

```
{
  tokenBaseCards(hash:"88bb7dc06faad55c77177446ed83047811234008") {
    label,
    expires,
    hash,
    customer_email,
    customer_id,
    profile_id,
    payment_id,
    method,
    active,
    created_at,
    updated_at,
    last_use,
    address {
      region {
        region_code,
        region,
        region_id
      },
      region_id,
      country_id,
      street,
      company,
      telephone,
      postcode,
      city,
      firstname,
      lastname
    },
    additional {
      cc_type,
      cc_last4,
      cc_exp_year,
      cc_exp_month
    }
  }
}
```

```
}
```

Example response:

```
{
  "data": {
    "tokenBaseCards": [
      {
        "label": "Discover XXXX-0012",
        "expires": "2021-03-31 23:59:59",
        "hash": "88bb7dc06faad55c77177446ed83047811234008",
        "customer_email": "roni_cost@example.com",
        "customer_id": 1,
        "profile_id": "1200144368",
        "payment_id": "1506360102",
        "method": "paradoxlabs_stripe",
        "active": true,
        "created_at": "2019-03-11 16:12:52",
        "updated_at": "2019-04-04 18:01:39",
        "last_use": "2019-04-04 18:01:39",
        "address": {
          "region": {
            "region_code": "MI",
            "region": "Michigan",
            "region_id": 33
          },
          "region_id": 33,
          "country_id": "US",
          "street": [
            "6146 Honey Bluff Parkway"
          ],
          "company": "",
          "telephone": "(555) 229-3326",
          "postcode": "49628-7978",
          "city": "Calder",
          "firstname": "Veronica",
          "lastname": "Costello"
        },
        "additional": {
          "cc_type": "DI",
          "cc_last4": "0012",
          "cc_exp_year": "2021",
          "cc_exp_month": "3"
        }
      }
    ]
  }
}
```

Fetch checkout config

Example request:

```
{
  tokenBaseCheckoutConfig(method:"paradoxlabs_stripe") {
    method,
    useVault,
    canSaveCard,
    forceSaveCard,
    defaultSaveCard,
    isCCDetectionEnabled,
    logoImage,
    requireCcv,
    sandbox,
    canStoreBin,
    availableTypes {key, value},
    months {key, value},
    years {key, value},
    hasVerification,
    cvvImageUrl,
    publicKey,
    country,
    currency,
    grandTotal
  }
}
```

Example response:

```
{
  "data": {
    "tokenBaseCheckoutConfig": {
      "method": "paradoxlabs_stripe",
      "useVault": true,
      "canSaveCard": true,
      "forceSaveCard": false,
      "defaultSaveCard": true,
      "isCcdetectionEnabled": true,
      "logoImage": "https://../images/logo.png",
      "requireCcv": false,
      "sandbox": true,
      "canStoreBin": false,
      "availableTypes": [
        {
          "key": "AE",
          "value": "American Express"
        },
        ...
      ],
      "months": [
        {
          "key": "1",
          "value": "01 - January"
        },
        ...
      ],
      "years": [
        {
          "key": "2019",
          "value": "2019"
        },
        ...
      ],
      "hasVerification": true,
      "cvvImageUrl": "https://../Magento_Checkout/cvv.png",
      "publickey": "pk_12345...",
      "country": "US",
      "currency": "usd",
      "grandTotal": 10000
    }
  }
}
```

Create card

Example request:

```
mutation {
  createTokenBaseCard(
    input: {
      expires: "2022-12-31 23:59:59",
      customer_ip: "127.0.0.1",
      customer_email: "roni_cost@example.com",
      method: "paradoxlabs_stripe",
      active: true,
      address: {
        region: {
          region_code: "PA",
          region: "Pennsylvania",
          region_id: 51
        },
        country_id: US,
        street: [
          "123 Test St.",
          "Apt 9"
        ],
        company: "",
        telephone: "111-111-1111",
        postcode: "12345",
        city: "Testcity",
        firstname: "John",
        lastname: "Doe"
      },
      additional: {
        cc_type: "VI",

```

```

        cc_last4: "0027",
        cc_exp_year: "2022",
        cc_exp_month: "12",
        cc_cid: "123",
        token: "ctoken_12345..."
    }
}
) {
    label,
    expires,
    hash,
    customer_email,
    customer_id,
    customer_ip,
    profile_id,
    payment_id,
    method,
    active,
    created_at,
    updated_at,
    last_use,
    address {
        region {
            region_code,
            region,
            region_id
        },
        region_id,
        country_id,
        street,
        company,
        telephone,
        postcode,
        city,
        firstname,
        lastname
    },
    additional {
        cc_type,
        cc_last4,
        cc_exp_year,
        cc_exp_month
    }
}
}

```

Example response:

```

{
  "data": {
    "createTokenBaseCard": {
      "label": "Visa XXXX-0027",
      "expires": "2021-03-31 23:59:59",
      "hash": "88bb7dc06faad55c77177446ed83047811234008",
      "customer_email": "roni_cost@example.com",
      "customer_id": 1,
      "profile_id": "1200144368",
      "payment_id": "1506360102",
      "method": "paradoxlabs_stripe",
      "active": true,
      "created_at": "2019-03-11 16:12:52",
      "updated_at": "2019-04-04 18:01:39",
      "last_use": "2019-04-04 18:01:39",
      "address": {
        "region": {
          "region_code": "MI",
          "region": "Michigan",
          "region_id": 33
        },
        "region_id": 33,
        "country_id": "US",
        "street": [
          "6146 Honey Bluff Parkway"
        ],
        "company": "",
        "telephone": "(555) 229-3326",
        "postcode": "49628-7978",
        "city": "Calder",

```

```

      "firstname": "Veronica",
      "lastname": "Costello"
    },
    "additional": {
      "cc_type": "VI",
      "cc_last4": "0027",
      "cc_exp_year": "2022",
      "cc_exp_month": "12"
    }
  }
}
}

```

Delete card

Example request:

```

mutation {
  deleteTokenBaseCard(hash:"88bb7dc06faad55c77177446ed83047811234008")
}

```

Example response:

```

{
  "data": {
    "deleteTokenBaseCard": true
  }
}

```

Place an order

Example request:

```

mutation {
  setPaymentMethodOnCart(
    input: {
      cart_id: "kDy0EKkjmIOxa6H3ceus6MjMasF9lao1"
      payment_method: {
        code: "paradoxlabs_stripe",
        tokenbase_data: {
          cc_type: "DI",
          cc_last4: "1111",
          cc_exp_year: "2022",
          cc_exp_month: "08",
          cc_cid: "123",
          token: "ctoken_12345...",
          save: true
        }
      }
    }
  ) {
    cart {
      selected_payment_method {
        code,
        tokenbase_card_id,
        tokenbase_data {
          cc_type,
          cc_last4,
          cc_exp_year,
          cc_exp_month
        }
      }
    }
  }
  placeOrder(
    input: {
      cart_id: "kDy0EKkjmIOxa6H3ceus6MjMasF9lao1"
    }
  ) {
    order {
      order_id
    }
  }
}

```

Example response:

```
{
  "data": {
    "setPaymentMethodOnCart": {
      "card": {
        "selected_payment_method": {
          "code": "paradoxlabs_stripe",
          "tokenbase_card_id": "88bb7dc06faad55c77177446ed83047811234008",
          "tokenbase_data": {
            "cc_type": "DI",
            "cc_last4": "1111",
            "cc_exp_year": "2022",
            "cc_exp_month": "08"
          }
        }
      }
    },
    "placeOrder": {
      "order": {
        "order_id": "000000255"
      }
    }
  }
}
```

Split Database

This module fully supports Magento Enterprise's split database feature (as of Magento 2.4; deprecated and subject to be removed in the future). No special setup should be necessary to get it working in a split-database environment. If you encounter any problems, please let us know.

Support

If you have any questions not covered by this document, or something isn't working right, please open a ticket in our support system: support.paradoxlabs.com

Support Policy: <https://store.paradoxlabs.com/support.html>

License and Terms of Use: <https://store.paradoxlabs.com/license.html>