Recurly | Get ahead of what's next

Integration Best Practices

Recurly Onboarding

Integration Best Practices

This document provides an overview of best practices for a technical integration with Recurly. Please feel free to reach out to the Recurly Onboarding team with any questions you may have while you configure your integration. Additionally our support team is available 24/7 to assist if an urgent issue arises and can be found using the methods referenced here.

Webhooks

Send webhook response receipt immediately

Recurly follows industry standards and does not recommend using webhooks as the source of truth. Webhooks may be sent out of order, retried or sent multiple times due to failed delivery or other server related errors. Please, ensure that your endpoint can receive the same notification multiple times *and* notifications that may arrive out of order.

Before processing a webhook, send Recurly a success response. This will prevent Recurly from resending a webhook. After sending the response receipt, you can process the webhook.

Don't rely too heavily on webhooks triggered by POST requests you sent to Recurly

The API is the most reliable data source. When creating data objects, such as subscriptions, use the POST API responses to take actions rather than waiting to receive certain webhooks. The POST request response has all data associated with the object created, whereas the webhook only contains the uuid needed to retrieve the object through the API.

Webhook payloads can be in JSON or XML formats. Each webhook endpoint can be configured to receive one payload type or the other. Recurly highly recommends the JSON webhooks due to their lightweight nature and alignment with best practices.

> *Example: When you create a new purchase, you'll get a response containing subscription, invoice, and transaction details. It's preferable to use the API response data rather than relying solely on webhooks for the new subscription, invoice, and transaction.*

Always make an API call when you receive a webhook that you take action on

Use the receipt of a webhook to trigger an API query to validate the push notification details against the current API data.

> *Example: Make a GET invoice API call after receiving a past due invoice webhook to confirm invoice details before taking an action like deprovisioning or sending customer communication.*

Developer Reference || Recurly Docs

API

Use the Latest version of the API (current version v2021-02-25)

To make use of our newest features please make sure you are using the latest version of the API / client libraries

Quick Start

Use as few API calls as possible to complete a checkout flow

Nest API calls wherever possible to reduce friction and increase efficiency.

> *Example: A common use case would be during a new account signup to send the account info, billing info and subscription info at the same time to the purchases endpoint.*

Developer Documentation

API Error Codes || Gateway Transaction Error Codes

Minimize GETs on page loads

The best practice is for you to set up a webhooks listener, receive webhooks about changes to those resources, and ping the API once each to update those details in your database. Now when the user goes to their account management page the details are

available in the database. They are current, load quickly, and don't require any extra API calls.

> *Example: Instead of checking what plan a customer is subscribed to when they log in or try to access content, store that information in your own database.*

Use cached plan data in the checkout flow

You should not need to GET plan details for every signup. Caching this data is the best practice.

> *Example: When a prospective customer is looking at your sales page, it is preferable to display cached plan information rather than hitting the Recurly API every time.*

Use filters when hitting list endpoints

Pass in filters such as begin_time and end_time to limit how many resources you are cycling over. If you're doing this for data warehousing, consider automated exports instead.

> *Example: Instead of fetching all invoices on an account, fetch just the invoices you're interested in such as past-due or created in the last 3 months.*

Sorting and Filtering Documentation (See Getting Started >> Pagination)

Use the Purchases endpoint

Transaction and Subscription POST endpoints will be deprecated in future versions of the API; you should leverage the Purchases endpoint in your workflows instead.

The exceptions are when creating future subscriptions or one-time charges that should be billed at a future date or when using Braintree tokens.

Purchases Endpoint API Documentation
Purchases Endpoint Documentation

Subscription Management

The Create Subscription Change endpoint is used to complete all of the upgrade and downgrade actions, and can specify the timeframe when you want that change to take effect. If choosing `bill_date, renewal, term_end,` it will put the subscription in a

pending state otherwise, the change will take place immediately and prorate the charges. Changes to ramps on active subscriptions can only be done immediately.

Subscription Change Documentation
Subscription Management Guide

Automated Exports

Use Automated Exports for data warehousing

Our Automated Exports feature is a much more efficient and reliable way to get data than cycling through resources in the API.

> *Example: Instead of paginating through all your accounts to see if any have changed, download the accounts export once per day from our Automated Exports endpoint. Use this CSV to process any account changes.*

Use Automated Exports to track events triggered in the last 24hr

Missed or delayed webhooks can cause your system to fall out of sync with Recurly. This can cause end-customer states to be inaccurate. Automated Exports are a great tool to use as a backup data source to ensure systems are never more than 24 hours out of sync.

> *Example: If it is mission-critical to store every Recurly invoice in your order management system, use the invoices export as a backup data source.*

Automated Exports Documentation.
Automated Exports API

Recurly.js

Don’t self-host or modify the Recurly.js library

The Recurly-hosted version of recurly.js is designed and regularly updated to maintain compatibility, compliance and security. Locally hosted versions of recurly.js run the risk of encountering issues with system interaction and incompatibility which may result in avoidable service interruptions on your client page. It is for this reason, we highly recommend against and do not support locally hosted copies of recurly.js.

We’ve prepared a full suite of example integrations for Ruby, Node.js, Python, and PHP using popular web frameworks for each language. These examples demonstrate the simplest method of integration, with a no-frills UI and are not meant to be set onto a web server and forgotten. They are intended to act as a suggestion, wherein your custom needs may fill in any implementation gaps.

Recurly.js Documentation

Recurly.js Integration Example

Client library releases:

- Node.js
- Python
- .NET
- Ruby
- Java
- PHP
- Go

3DS Secure Checkout

When discussing 3DS (Three-Domain Secure) authorization, especially in the context of online payments, here are some key points developers should know:

What is 3DS?

Three-Domain Secure (3DS) is an authentication protocol designed to enhance the security of online card transactions by adding an additional verification step for cardholders. The term "Three Domains" refers to the three parties involved. The issuing bank (the cardholder's bank), The acquiring bank (the merchant's bank), The interoperability domain (the payment network, like Visa or Mastercard).The 3DS process may introduce an extra step in the checkout process. Make sure to design this flow to minimize friction for users, while still ensuring security.

3DS Versions

- **3DS 1.0**: The original version, which required a password or OTP (One-Time Password).
- **3DS 2.0**: Introduces improved user experience with support for mobile devices, biometric authentication, and a better data exchange between parties.

If you are planning to do business in the European market then you'll need to ensure 3DS is part of your checkout flow in order to comply with PSD2 regulation by providing multi-factor authentication for online transactions where a consumer is in session. A guide on implementing 3DS as part of your integration can be found by following this link to our Dev Docs. Additional 3D Secure documentation can be found here.

When you are ready to test the integration with your payment gateway, contact the Recurly support team or your Enablement Manager for assistance in configuring your site in 'Development' mode. You'll need to contact your gateway in order to obtain their specific set of test credit cards for 3DS2. Please also see our guide on gateway specific configurations here.

Testing Considerations

Testing is a critical component of your preparation for launch. Think about your customer lifecycle and perform end-to-end tests for every permutation of that journey to ensure that you achieve your expected results - both customer-facing and internal.

This document offers some suggestions for your testing process. Due to the wide variety of business models and use cases we see, as well as the various external systems that may be integrated within a company's ecosystem, this document should not be considered comprehensive. The goal of this document is to help you consider your testing strategy and offer some examples of what to look for.

General API and Recurly.js Functionality

- Are you running the most recent version of the API?
- Do you have a designated EU Recurly site that uses the Recurly EU data center? If so, are you using the EU url for api endpoints?
- Do you get the desired responses from the API calls you use?
- Do you use the most efficient api calls to fulfill your use case (e.g. purchase endpoint vs create account, create subscription endpoints)?
- Do you streamline API calls where possible, to increase efficiency? (e.g. in a checkout flow, you can use certain endpoints such as the purchase endpoint to make one API call to accomplish multiple tasks: create an account, add billing info, create an adjustment, and create and post an invoice)
- Do you have proper messaging in place for API response errors?
- Do you have proper messaging in place for Recurly.js input errors?
- Do you have the appropriate transactions 3ds2 challenges mechanisms in place using Recurly.js?

Webhooks

- Do you receive and store all of the webhooks you need, at the correct endpoints?
- Do those webhooks trigger API requests where appropriate/expected?
- Do you send a 200–series response back to Recurly for all webhooks, to confirm receipt and ensure we don't continue to resend them?

- Do you have your endpoints set up to receive the same notification multiple times and in the wrong order in case of automatic retries due to a failed delivery status?
- Do you take action on closed accounts via receipt of a closed account webhook? If so, you should verify accounts are still closed by issuing an API request.
- Do you avoid using webhooks to confirm POSTs and PUTs when you could rely on a 20X response

Automated Exports

Used for data warehousing. More efficient and reliable way to get data than cycling through resources in the API. Exports generally provide a broader set of attributes about each object than what is available via each API object.

- Do you have the correct exports set up?
- Are you able to get the expected exports via the API?
- Are you polling the specified endpoints approximately 2 hours after the exports have run to see if data is ready for download. If the export is not available for download, you will receive a 404 response until it is fully uploaded and available.
- Are you able to successfully run a diff between the export and your database?

Customer Experience

Validate your end-customer experience, throughout their entire lifecycle. Customers will interact with a sign-up and checkout flow, and you may allow them to log into your application to manage their accounts. Are they able to execute all desired actions (e.g. update billing info, upgrade or downgrade, change address, cancel their subscription, etc.)? Do your other systems receive and process these updates accordingly?

- When a customer signs up / checks out
    - Are the Web vs Mobile experiences consistent?
    - Is the experience across various browsers consistent?
    - Is it easy for them to see/choose their payment option?
    - Do purchases across all supported payment methods function properly? (e.g. credit card, Direct Debit, PayPal, Amazon Pay, Apple Pay on the Web, etc.)
    - Is the 3DS2/SCA challenge being triggered (if used)?
    - At a failed signup attempt, do they see useful error messaging?

    - If using account hierarchy, is the correct parent-child relationship being created? Is the child or parent correctly listed as “bill to”?
    - Are the correct addon tiers being used?
    - Are they able to enter a coupon code or multiple coupon codes?
    - Do the coupons apply discounts as expected?
    - Are they able to redeem a gift card?
    - Are taxes previewing correctly?
    - Are tax rates correct for your various tax-enabled regions?
    - Is the correct address being used to calculate taxes (account vs billing address vs shipping address)?
    - Do they receive all expected emails?
    - Is the content of those emails all customized and branded properly?
    - Does the invoice read as expected (customer notes, T&Cs, line item descriptions, etc.)?
    - Do other systems receive and process all necessary “new customer” information provided by Recurly? (e.g. is service/access provisioned, do you display an in-app “welcome” notification or purchase confirmation, etc.)
- When a customer makes subscription changes (upgrade, downgrade, pause, cancel, terminate, etc.)
    - Based on your invoice settings, do they receive an appropriate error message when upgrades/downgrades are blocked?
    - Is the subscription paused until the correct date/time?
    - Is the subscription expired as expected? (cancel = expire at renewal, terminate = expire immediately)
    - Is the correct credit or refund amount being applied if a downgrade occurs immediately?
    - If a credit is given, is this applied to the next invoice correctly?
    - Do they receive all expected emails?
    - Is the content of those emails all customized and branded properly?
    - Do other systems receive and process all necessary customer subscription updates?
- When a customer changes account or billing information
    - Are the updates applied correctly in Recurly?

    - Do they receive helpful error messages if an invalid update is applied?

- When a customer's subscription successfully renews
    - Do they receive all expected emails? This could include a trial ending email, payment confirmation email, etc.
    - Is the content of those emails all customized and branded properly?
    - Do applicable coupons process as expected?
    - Do free trials convert as expected?
    - Do other systems receive and process all necessary invoice status updates?
    - Do other systems receive and process all necessary subscription status updates?
    - Do all downstream processes function as expected (e.g. fulfillment, shipping, CRM or Finance system updates, etc.)

- If their renewal payment is declined
    - Do they receive all expected emails, and in the correct order, if you use multiple dunning email templates?
    - Is the content of those emails all customized and branded properly?
    - Does the link to update their billing info resolve to the correct URL?
    - Does their billing info update correctly?
    - Do they receive a helpful error message if not?
    - Do other systems receive and process all necessary invoice status updates?
        - If the invoice is paid initially
        - If the invoice goes past due
        - If the invoice is paid mid-dunning
        - If the invoice fails at the end of dunning
        - If the invoice remains past due at the end of dunning
    - Do other systems receive and process all necessary subscription status updates?
        - If the subscription is expired due to non-payment
        - If the subscription remains active after dunning completes without a successful payment
    - If you utilize dunning webhooks, are other systems triggered by those webhooks as expected, for each step of dunning?

    - Do all downstream processes function as expected (e.g. prevention of fulfillment and/or shipping, make updates to CRM or Finance systems, etc.)

- If they successfully make a one-time purchase using an existing account
    - Do they receive all expected emails?
    - Is the content of those emails all customized and branded properly?
    - Does the invoice display all information as expected?
    - Do all downstream systems receive the new purchase information as needed?

- If they attempt to make a one-time purchase using an existing account with billing information on file, but the payment is declined
    - Do they receive helpful error messaging?
    - Do they have a way to update their billing info and attempt the purchase again?

Upstream, Downstream or External Processes and Systems

Have you included all relevant downstream or external processes and systems in your integration? Examples might include:

- CRM & ERP systems
- Salesforce & CPQ applications
- Shipping and fulfillment software
- Accounting applications
- Data Warehousing applications
- Do all integrated systems receive and use information from Recurly as expected?
- Does your customer provisioning process trigger as expected?
- Does your customer deprovisioning process trigger as expected?
- Have you reviewed manual exports versus automated exports (if applicable)?
- Do you have reporting processes in place based on the data you want to use?

Payment Gateway / Transactions

- Do payment tests produce the expected transaction results for each gateway you use? (in Sandbox and Development mode; we recommend minimal Production testing as data created in Production is indelible)

- Do you have your site in Development mode to test specific gateways? Sandbox only uses the Recurly test gateway
- Do your settings in the Development site match your settings in the Production site?
- Do you need to have a different gateway setup for each of your different payment methods and/or currencies? Are they being routed properly? Are you using gateway_code to override defaults, if needed?
- Do the transactions look as expected? (in Recurly and in the gateway)
- For 3ds2, is the SCA challenge being triggered appropriately?
- Do you receive expected responses for both valid and invalid billing info across all supported payment methods?
    - Credit card
    - PayPal
    - ACH
    - Amazon Pay
    - Apple Pay on the Web
    - SEPA
    - BACS
    - BECS
    - Other

Billing Information	number	month	year	name_on_account	account_number	routing_number	account_type	sort_code	bsb_code	type	iban	billing_agreement_id
CC	x	x	x									
ACH				x	x	x	x					
BACS				x	x			x		x		
BECS				x	x				x	x		
SEPA				x							x	
Amazon												x
PayPal												x

Admin User Interface Functionality

- Add a User
    - Are user rights restricting access as expected? Do you need to modify access for anyone?
- Enable Two-Factor Authentication (2FA)
    - Do you have a process in place to ensure that all users of Recurly enable 2FA?
- Enable Single Sign-On (SSO)
    - Do you have a process in place to enable Okta SSO for your users?
- Deprovision a user
    - Do you have a process in place to remove access for users who no longer require access to Recurly?
- API Access Configuration
    - Do you need more than one API key? Do you have a process in place to generate Private API Keys when needed?
    - Do you have a process in place for revoking API keys?
    - Do you need to test a read-only private API key?

Customer Team Interactions with the User Interface

Ensure your team is trained to apply the proper changes that fit your business needs. The following are common actions taken by customer facing users of Recurly. You may want to ensure that you have policies in place for how each action should be handled.

Accounts

- Are you able to create an account with all the needed fields, custom fields, billing info and address fields? Are you able to create parent-child accounts (if hierarchies are enabled)?
- Are you able to edit existing accounts and billing info?
- Are any one time charges correctly added to an account and transacted?
- Is the correct tax being calculated for the subscription, items/one-time fees, addons and/or setup fees?
- Are discounts, coupons or free trials correctly being added to a subscription, account, or one-time fee?

- Is account closure appropriately canceling the subscription and removing the billing details?
- Are you able to add credits to the account?
- Are you able to add account notes?
- Are you able to use hosted pages (e.g. provide url to edit billing info)?

Subscriptions

- Are you able to add the subscription to the account with the correct renewal dates, term dates or contract dates? Are subscription add-on tiers charging at the right tier level?
- Are setup fees being charged?
- Subscription Actions
    - Edit Subscription
        - Apply Changes Immediately, Next Bill Date or When Terms Renew
    - Add Item/Charge
        - Invoice Now or At Next Bill Date
        - Open Amount or Item
    - Add Credit
    - Change Next Bill Date
    - Pause Subscription
    - Cancel Subscription
        - Full, partial, no refund

Invoices

- Invoice Actions
    - Download PDF
    - View Hosted Invoice
    - Edit Invoice
    - Issue Refund
        - Quantity or Specific Amount