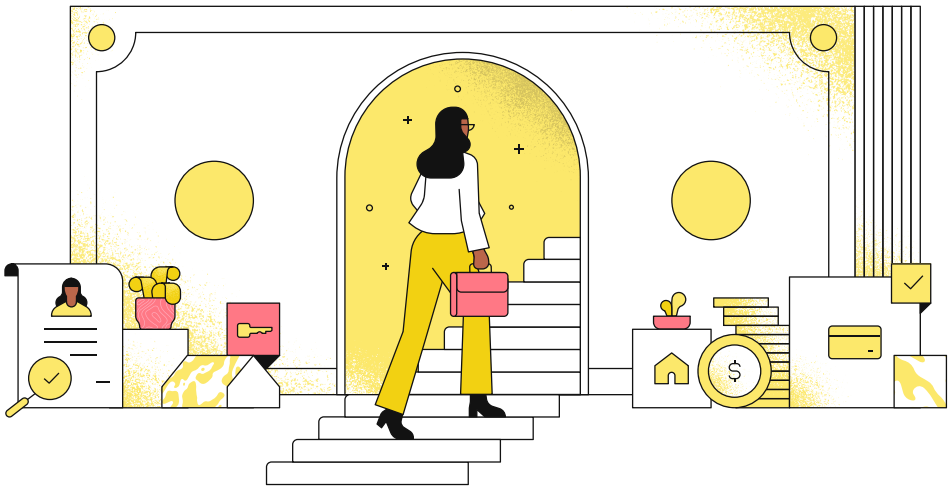
PLAID

plaid.com

Plaid implementation handbook

Contents

We put together this guide to help you progress towards a successful production launch as efficiently as possible. We designed our API with simplicity and developer experience in mind, but the complex nature of banks and financial data means that there are still many factors you must take into consideration when building out your Plaid integration and user experience.



Get started

Sign up for API keys

First you will need to create an account to access your Plaid dashboard. We recommend having one centralized dashboard for your team as this will allow you visibility into all associated activity.

Verify company contact information by going to Team settings > Company to ensure any official correspondence (including billing) from Plaid addresses the correct entity/stakeholder.

- Company details and location information is not shared with users of your app and is only accessible to Plaid, members of your team, and financial institutions you register with.
- Technical contact and billing contact: This information is private and is not shared with users of your apps. This is only accessible to members of your team and Plaid.

Find your API keys by going to Team settings > Keys: client_id (Plaid-specific identifier for your company) and secrets (secrets differ for each environment) are used to access data through the Plaid API.

Add users

Provide all of your teammates access to your Plaid dashboard by going to Team settings > Members. Various permissions can also be set on a per-user basis, depending on their role and needs. When adding a teammate, they will be sent a verification email that they will need to confirm before being able to access the Plaid dashboard.

If you are looking to expand across business lines with multiple use cases, you can create a new team to generate a fresh set of API keys without having to create a new dashboard login.

Note: Please work with your account manager on this as it does have billing implications.

Dashboard management

Go to Activity > Usage to be able to view high level product usage month over month once you start making requests to our API.

Note: Currently, only development and production activity is available in "Usage."

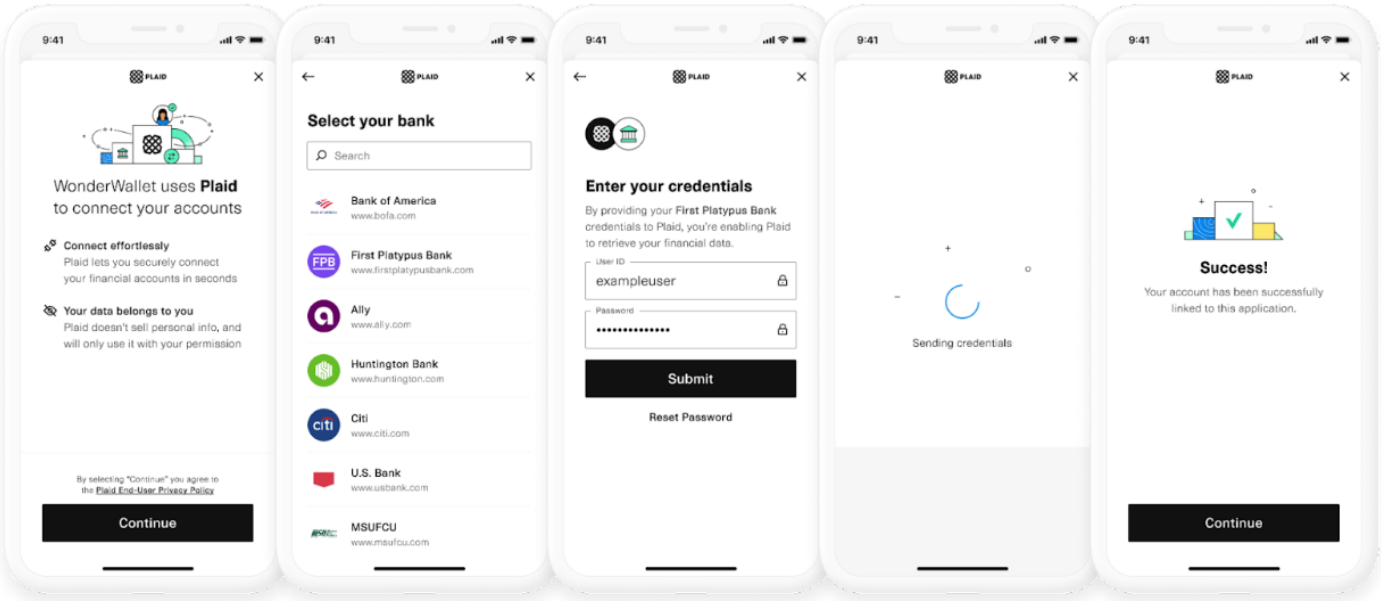
Go to Activity > Status to view a summary of institution health for Plaid supported financial institutions. The list will default to your top 18 institutions (based on what you establish in the customize portion of your dashboard for Link) and you can search by FI name as well. You are also able to go a level deeper to view institution errors on a product basis.

Go to Activity > Logs to view in-depth details on requests made to our API including: Descriptions, institutions, environments, timestamps, and responses for most recent 14 days of activity.

Note: While we try to provide as much useful data as we can, many customers supplement the activity module in their Plaid dashboards by building custom reporting of their own (using Tableau, Periscope, or any comparable tool).

Implement & configure Plaid Link

Link is the primary interface through which a user will interact with Plaid. Users create Items (an Item is a Plaid term representing a login at a financial institution) by entering their credentials into Link, which allows Plaid to establish the connection and provide the data to power your application.



User experience pre-Link

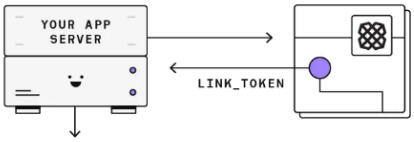
Consider the following recommendations and UI/UX best practices to boost conversion and create a first-class user experience:

- Provide a value proposition to clearly explain why the user should link their account. Consider the benefits of speed (instant verification) and cost (free to use).
- Introduce Plaid Link and inform the user that you use Plaid and that they will be leaving to a new experience. Avoid surprises and prime the user before they enter the Link experience.
- Design for your user by using tags such as 'recommended' or 'preferred' to steer users to the ideal flow. Good design increases curiosity and trust to motivate users to convert.
- Highlight security and transparency by including a lock icon, mentioning '256-bit encryption', or that credentials are never stored by your application.

Link token exchange

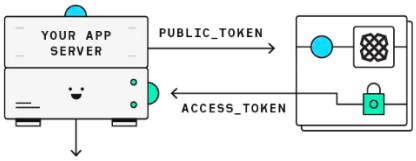
The Plaid flow* begins when your user wants to connect their bank account to your app.

1 Call /link/token/create to create a link_token and pass the temporary token to your app's client.



2 Use the link_token to open Link for your user. In the onSuccess callback, Link will provide a temporary public_token.

3 Call /item/public_token/exchange to exchange the public_token for a permanent access_token and item_id for the new Item.



4 Store the access_token and use it to make product requests for your user's Item.

*Please note that the token exchange flow might vary depending on your products and use case (i.e., Income).

Refer to the full Plaid documentation to ensure the correct flow.

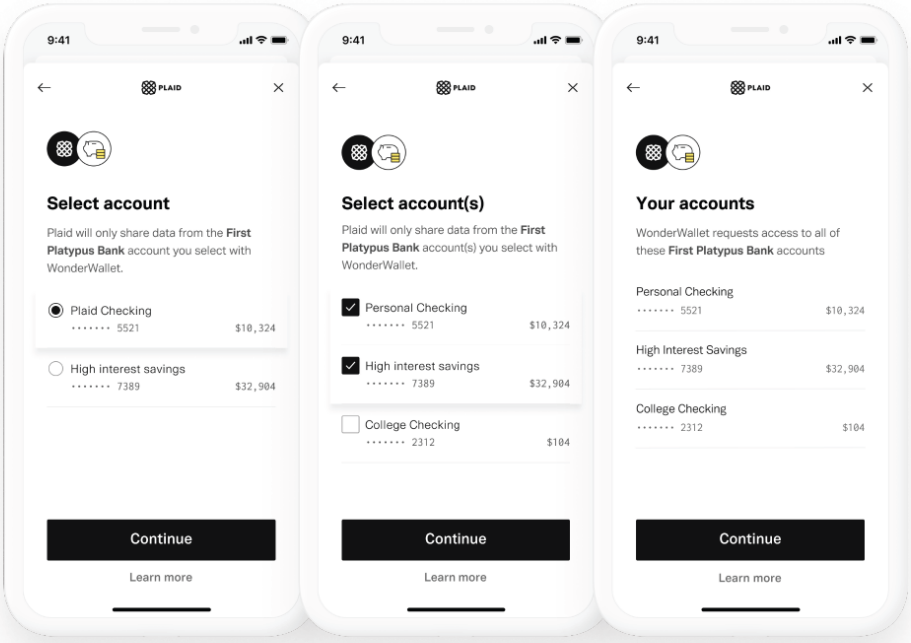
Initializing Link

Identify products initialized during Link:

This ensures Link shows the correct set of institutions based on your use case/need. You can initialize Link with one or multiple products (Note: Balance cannot be initialized via Link as it requires an existing Item). Only institutions that support all initial products will be displayed. In other words, if you require Auth and Identity for all users, you'll set the product array to ['auth', 'identity']. However, if you only require Auth and use Identity as an add-on product, then you'll set the product array to ['auth'] only. You'll then separately request Identity afterwards for institutions that also support it.

- For a full list of Plaid products and use cases, please see here.

When considering initializing Link with multiple products, please remember that if a recurring product is used such as Transactions, the recurring product on the created Item cannot be used one-time or "disabled" later without entirely removing the associated Item.



ENABLED FOR ONE ACCOUNT

ENABLED FOR MULTIPLE ACCOUNTS

ENABLED FOR ALL ACCOUNTS

Link configuration

Link supports a number of customizations through the dashboard, including, but not limited to: Changing the background color/opacity on desktop, reordering institutions featured in the “Institution select” pane, and customizing the copy across several other panes.

We also support multiple configurations of Link for your app. Easily customize the account sign-in flow based on your user’s location, the language they speak, features they’re using, and more. Use the link_customization_name parameter in your initialization code to point to a specific configuration.

Select account

The “Account select” pane is intended to give users control and transparency into the accounts they share with you and Plaid. Plaid will only provide the data for those accounts selected by the end user. There are three options for configuring account selection:

1. Enabled for one account
2. Enabled for multiple accounts
3. Enabled for all accounts

When new accounts are discovered for existing Items, Plaid will fire a NEW_ACCOUNTS_AVAILABLE webhook. Prompt the end user to go through update mode to permission the new account(s). Set the update account_selection_enabled flag to ‘true’ when calling /link/token/create.

Account filters

Account filters allow you to select which account types and subtypes will be available in Link. They can be configured via the account_filters parameter when creating a Link token.

Account types and subtypes that are not compatible with the products used to initialize Link will be automatically omitted and do not require an account filter setting. For details on account type and subtype compatibility, see the Account type / Product support matrix.

Update mode

Over time, Items may need to refresh authentication information. This can happen if the user changes a password, if MFA requirements change, or if the login becomes locked. Typically this is indicated by receiving an ITEM_LOGIN_REQUIRED error, an ITEM: ERROR webhook, or a PENDING_EXPIRATION webhook.

To use update mode for an Item, initialize Link with a link_token configured with the access_token for the Item that you wish to update. Following completion of update mode, the existing access_token and item_id will remain the same.

Note: No products should be specified when creating a link_token for update mode.

To test update mode in the sandbox, use the /sandbox/item/reset_login endpoint to force a given Item into an ITEM_LOGIN_REQUIRED state.

It is important to convince and guide users to re-Link through update mode. When doing so, please consider the following options:

Notifications

- Present a new screen: Ensure that the user becomes aware of the situation.
- Communicate the problem visually: Expressions, warning signs, or icons will communicate to the user that they must take action.
- Provide pathway to fix: Clearly guide the user to the update mode Link flow.

In-app banners

- Contrast colors: Present the message in a contrasting, cautionary color which will clearly grab the user's attention.
- Clear messaging: Explain why the user is receiving this notification, and what they can do to fix the error.
- Define the call to action: Position the alert in a way which prompts the user to take immediate action.

Dashboards

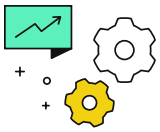
- Alert users: If they regularly log on to your application, place alert icons and/or messages throughout the screen to create a new path for the user to address in order to return functionality to a normal state.
- Fade coloring: Changing the unlinked account, card, etc. to a gray tone indicates to the user that it is currently unavailable.

Text messages

- Provide value proposition: Present compelling reasons and benefits to the functionality gained by re-linking an account.
- Persist in communication: Users sometimes forget to take an intended action, so providing consistent and well-timed reminders can help prompt eventual action.

Emails

- Communicate the full message:
  - Why is the user receiving the email? (i.e., "reconnect your bank account").
  - What might have happened to cause this error? (i.e., "it looks like your bank login has timed out").
  - How will this error affect the user's ability to receive this service? (i.e., "if this issue persists, we will not be able to see the latest transaction activity associated with this account").
  - What actions can the user take now to resolve the situation? (i.e., call to action with button to launch update mode).



Build a robust backend

Leverage SDKs

- Plaid SDKs for iOS, Android, and React Native allow for simplified, comprehensive integration of Link into mobile apps with dedicated support from Plaid.
- Plaid offers official API libraries for different programming languages (Node, Java, Ruby, Python, Go), which are regularly updated for breaking and non-breaking API changes.
- New versions of SDKs may be released via monthly updates with the latest features: Minor and major releases with bug fixes and new feature support are rolled out monthly to keep your mobile app fully up to date.
- Plaid does not force updates, but we do recommend maintaining as current a version as feasible.
- For OAuth flows, we strongly recommend the use of our SDKs, as they abstract away a lot of the work required to implement OAuth.

Handle callbacks

Are you appropriately handling Link callbacks?

Handle callbacks beyond just onSuccess in order to gracefully handle errors and build analytics around Link.

- Listen to the onExit() and onEvent() callbacks for error_type and error_code in order to implement error handling.
- Listen to the onEvent() callback for exit_status or timestamp in order to implement Link conversion analytics.

Link's onSuccess callback is called when a user successfully links their account. This is where Link hands off a public token that you will exchange for an access token server-side. The onSuccess callback is also where you will know which account was selected (in the metadata) if you have enabled the account selection feature in your Plaid dashboard. Otherwise, the onExit callback is called when a user exits Link without linking their account. You may also want to use the metadata provided in this callback as well as the onEvent callback to run analytics on end-user behavior through your analytics platform of choice (Periscope, Tableau, or any comparable tool). For example, in the case where the onExit callback contains a metadata status of "institution_not_found".

Log identifiers

Are you logging and storing all relevant API identifiers (access_token, item_id, link_token, request_id, account_ id, link_session_id)? Note that different Plaid products might have additional key identifiers for logging (i.e., user_token and user_id for Income and asset_report_token for Assets).

- It's important to properly store Plaid identifiers returned by Link and direct API endpoints. Plaid identifiers let you associate API and institution events with your requests, and will help our Support team resolve your issues faster and more accurately.

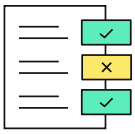
Log Plaid identifiers and IDs properly to enhance security, when contacting Support about a specific request or callback, and for finding specific entries in the activity log.

- Access tokens and Item IDs are the core identifiers that map your users to their financial institutions. Store them securely and associate them with users of your application. Make sure, however, that these identifiers are never exposed client-side. Keep in mind that one user can create multiple Items if they have accounts with multiple financial institutions.
- Ensure that the following identifiers are securely logged, as they will be needed when contacting Support about a specific request or callback.

link_session_id: Included in the onExit, onEvent, and onSuccess callback of a Link integration.

request_id: Included in all Plaid API responses.

account_id: Included in all successful Plaid API responses that relate to a specific Item or account.



Handle errors

It is critical to be aware of the different types of errors you can encounter, how to communicate certain errors to end-users, and what actions you can take to resolve these errors. Errors most commonly occur at the Link / Item, API, or Institution level. For a full breakdown of all error types and troubleshooting steps, see our documentation.

Link errors

- In most cases, Link will guide your user through the steps required to resolve the error.
- The majority of Link errors are user actionable such as INVALID_CREDENTIALS or INVALID_MFA.
  - » For further Link debugging for common issues such as a missing institution, unhealthy institution, missing accounts, or issues with OAuth FIs, refer to the Link troubleshooting documentation.
- All Link errors should be logged from the onEvent and onExit callbacks for support teams to reference.
- Use the Item debugger in the Plaid dashboard to see recommended steps to resolve.
- All institution errors can be cross referenced on the “Institution status” page in the Plaid dashboard.

API errors

- API errors should be logged with the request_id in the response from Plaid.
- Use the Item debugger in the Plaid dashboard to see recommended steps to resolve.
- All institution errors can be cross referenced on the “Institution status” page in the Plaid dashboard.
- ITEM_LOGIN_REQUIRED should programmatically trigger Link in update mode for the user to resolve the issue.

Item errors

Are you set up to handle Item errors?

Occasionally, due to credential changes, security updates, institution infrastructure issues, or other factors, an Item can enter an error state that renders it inaccessible to Plaid. When this happens, we'll either fire an Item error webhook or you will hit an error in response to calling our API endpoints. Note that you can also check an Item's state at any time through the /item/get endpoint. To re-establish an Item's connection to an institution, the user needs to re-authenticate in Link's update mode. To launch Link in update mode, generate a new link_token configured with the access_token for the Item that you wish to update.

End user error messaging in Link

The most common errors encountered in Link are likely user actionable errors. Link will proactively notify users of the specific error and required action (i.e., re-submit credentials or MFA, try another bank, etc.).

Rate limits

Errors of type RATE_LIMIT_EXCEEDED will occur when the rate limit for a particular endpoint has been exceeded. Default rate limit thresholds for some of the most commonly rate-limited endpoints are shown below. Note that these tables are not an exhaustive listing of all Plaid rate limits or rate-limited endpoints, that some users may experience different rate limit thresholds from those shown, and that rate limits are subject to change at any time. To see the rate limits for Plaid environments, see here.

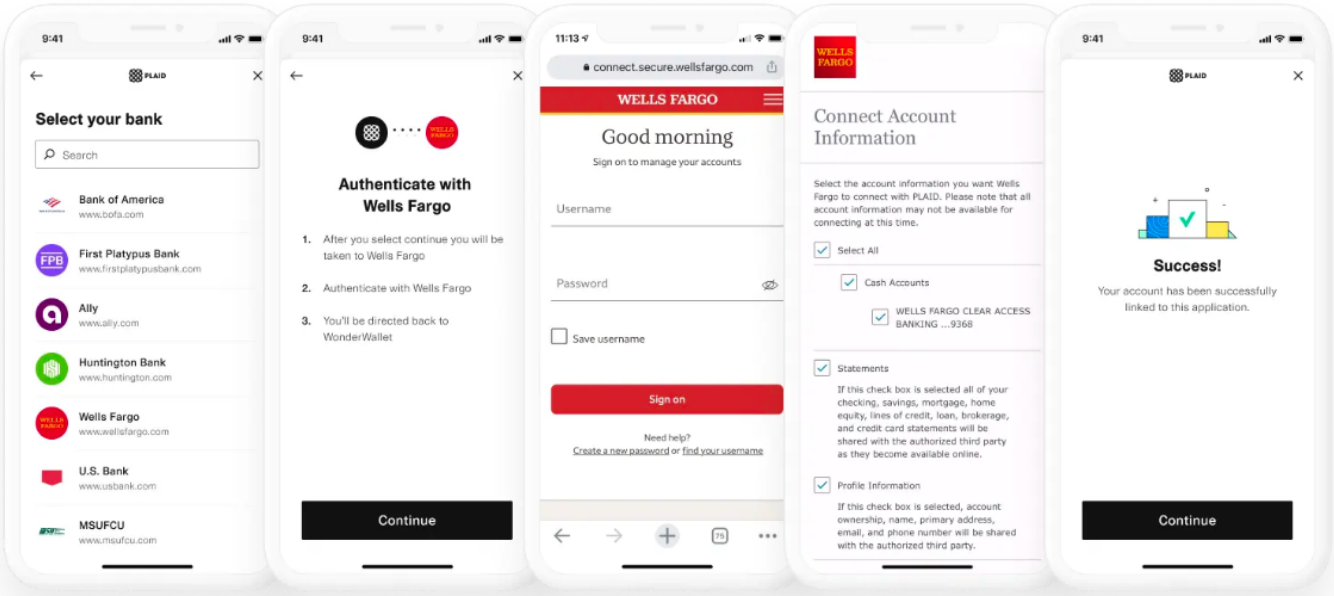
Ensure OAuth support

OAuth framework

OAuth is an industry-standard protocol for authenticating and permissioning data to third parties. With OAuth, end users can grant third parties access to their data without sharing their credentials directly with the third party. With Plaid's OAuth integration, users will be directed to the bank hosted website or app to complete their sign-in, before returning to Link.

Note: Some major institutions will only be accessible through OAuth.

- To increase user confidence in connecting their accounts, we direct the user to their financial institution's website to authenticate with their credentials.
- In mobile apps a user can be directed to the institution's app to complete their sign-in using credentials or biometric data.
- Once the authentication flow is complete, Plaid is able to retrieve account data directly from the institution via our direct API.



Implementation approach

- **Prerequisites:** Link tokens, bank addendum, security questionnaire, dashboard application profile completed.
- **Redirect URI:** Configure a landing page for Link to relaunch from after the OAuth flow is complete.
- **Link configuration update:** Based on your Link integration type, adjust the code for initialization and relaunch, mobile SDK's are recommended.
- **Persist user state:** Store Link token and metadata to continue flow after relaunch.
  - » Depending on your client platform, Link may require additional configuration to work with OAuth. See here for platform-specific details.
- **Institution specific details:** Make technical updates or process considerations for product API calls.
- **Link callbacks:** Include new onEvent metadata in conversion metrics.

Redirect URI

All OAuth Link integrations require a redirect URI to be established in the code and on the Plaid dashboard.

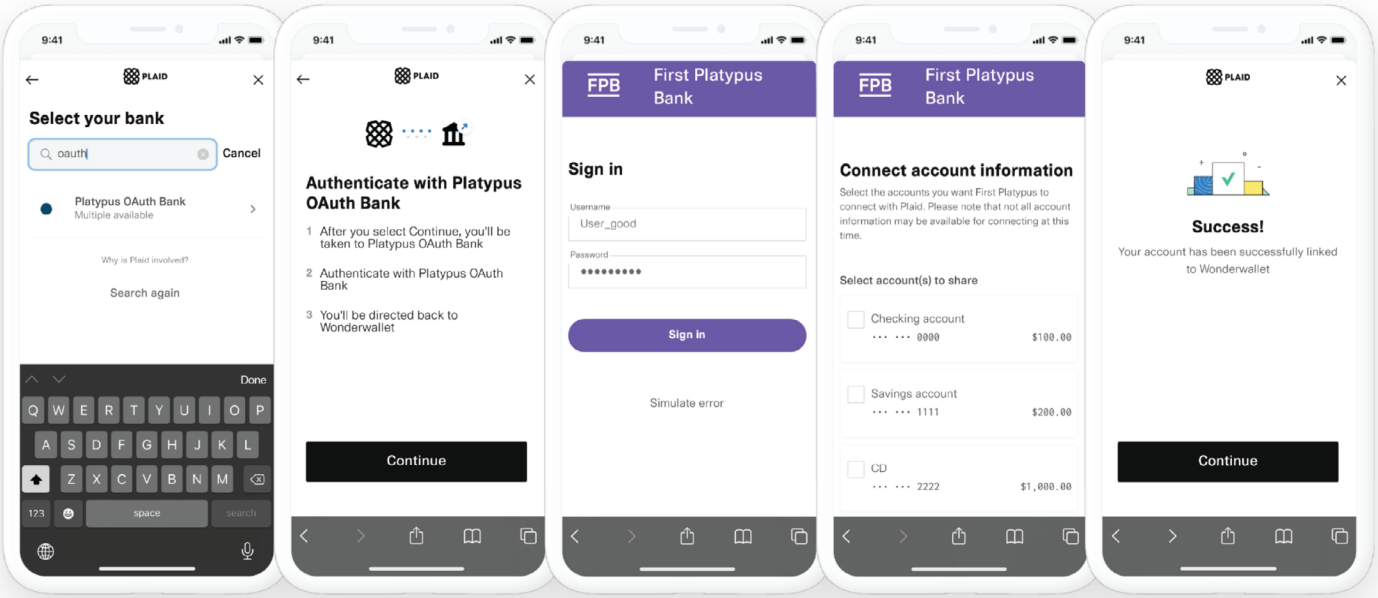
- On the Plaid dashboard, add redirect URIs that your app(s) will use to relaunch Link after the OAuth process is complete.
- URIs should be generic and not contain any query parameters as the redirect will amend the URI with OAuth data.
  - » Example: https://www.myapp.com/oauth-redirect-plaid.html.
  - » For Android (Native SDK or RN Android SDK): Create and store the Android Package Name properly in the Plaid dashboard.
  - » For iOS: Configure your redirect URI as an Apple App Association File.

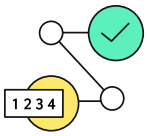
Testing OAuth

Platypus OAuth Bank (ins_127287) and Platypus OAuth App2App Bank (ins_132241) allow you to simulate OAuth in sandbox.

- Setup the redirect URI to be included in the link/token/create call and specify it on the Plaid dashboard.
- **Create a Link token in sandbox** using the redirect_uri or android_package_name & search for **Platypus OAuth Bank** in the "Institution select" pane on desktop.
- For mobile apps, test with **Platypus OAuth App2App Bank.**

Note: Custom sandbox users are not supported with this institution.





Listen to webhooks

When to use webhooks

Are you initializing Link with a webhook URL?

- Webhooks allow you to get notified when new data associated with an Item is available for retrieval. Webhooks will also be sent when an Item encounters an error due to changes to user credentials, MFA, or security settings (you'll then prompt the user to re-authenticate in Link's update mode).
- Specific products that utilize webhooks are: Transactions, Auth (additional flows beyond Instant Auth), Assets, Investments, Liabilities, and Income.

Listening for webhooks

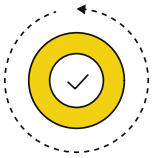
- Plaid sends POST payloads with raw JSON to your webhook URL from these IP addresses. Be sure to whitelist these IP addresses for receiving webhook requests.
- If there is a non-200 response or no response within 10 seconds from the webhook endpoint, Plaid will retry sending the webhook up to two times with a few minutes in between each webhook.

Post-webhook logic

- For use cases where Plaid is retrieving additional data, i.e., Transactions updates, receiving webhooks should serve as a call for additional action.
- Webhooks with code ITEM_LOGIN_REQUIRED should prompt the user to go through Link in update mode.
- Webhooks with specific product codes such as DEFAULT_UPDATE for Transactions, should trigger a call to transactions/get.

Webhook verification

While webhooks do not contain any sensitive information, Plaid provides an extra optional layer of security via webhook verification. Plaid signs all outgoing webhooks so you can verify the authenticity of any incoming webhooks to your app. The verification process requires understanding JSON web tokens and JSON web keys.



Test your solution

Once your core integration is completed, it will be important to test extensively to optimize your workflows and uncover any edge cases. We recommend doing most of your testing in our sandbox environment, followed by final tests in our development environment prior to launching.

Sandbox testing

Our sandbox environment uses dummy data. There are no Item-add limits and unlike our other environments, you can create Items programmatically without using Link for testing. In sandbox, you have the ability to customize certain behavior and traits of Items you create to simulate certain scenarios or "user profiles." You also have the ability to force error codes and fire on-demand webhooks.

Development testing

Our development environment uses live data. Real credentials are used to link real bank accounts in this environment. While we've tried to make our sandbox environment as comprehensive as we could, there are always edge cases that can only be uncovered using real data.

Note: You have a limit of 100 Item-adds in development that cannot be transferred to production (removing an Item does not reverse/reset the count).

Custom users

The sandbox environment provides the ability to create custom user accounts, which can be used in conjunction with /sandbox/public_token/create or Plaid Link to generate custom sandbox data, or in conjunction with Plaid Link to test Link flows in the sandbox.

Developers can create and save a user with the custom_'suffix' username that's tied to a specific data with the JSON. Then, they can simply input the custom_'suffix' username and any non-null password.

Set up and configure your custom users JSON via your dashboard. Note that there is a limit of 100 custom users per client_id.

TESTING CHECKLIST

- [ ] **Sandbox**

  The default username/password combination for all sandbox institutions is user_good / pass_good. Note that the password will change when triggering errors in sandbox - see “Errors” section below.

- [ ] **Callbacks**

  Listen to the onExit() and onEvent() callbacks for error_type and error_code in order to implement error handling.

- [ ] **Link in update mode**

  When a user changes their credentials, MFA, or security settings, or when a bank makes changes to its infrastructure, Plaid’s connection to the user’s bank account may be disconnected, which results in an ITEM_LOGIN_REQUIRED error. In order to resolve the error, you’ll need to prompt the user to re-authenticate their account using Link’s update mode.

  Force a sandbox Item into an error state by calling /sandbox/item/reset_login/ to test Link’s update mode flow in sandbox.

- [ ] **Webhooks**

  Ensure you have configured your webhook URL via the webhook parameter of /link/token/create when adding an Item.

  Plaid provides the /sandbox/item/fire_webhook endpoint that can be used to trigger a DEFAULT_UPDATE webhook, allowing you to test that you are receiving webhooks successfully.

- [ ] **Errors**

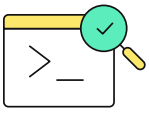
  In the sandbox environment, you can trigger a number of ITEM_ERROR or INSTITUTION_ERROR errors that you could typically receive when creating an Item. Do so by using username user_good and modifying the password: error_[ERROR_CODE].

  For example, the password error_ITEM_LOCKED will simulate an ITEM_LOCKED error.

  The list of errors that can be triggered in sandbox can be found here.

- [ ] **Logging**

  Ensure you have appropriate logging and storing infrastructure for all relevant API identifiers (access_token, item_id, link_token, request_id, account_ id, link_session_id, etc.) as this is very important for identifying, troubleshooting, and resolving issues.



Prepare to go live

Request production access

Request production access via your Plaid dashboard. It can take 1-2 business days to process this request. Once approved, your production keys will be available in Team settings > Keys.

PRE-GO LIVE CHECKLIST

- ☐ **Initializing Link**

  Make sure you are calling /link/token/create with the right parameters (products, country codes, webhook url, etc.).

- ☐ **Institution handling**

  Using callbacks to get institution_id.
  Calling institutions/get_by_id to check the status and health of the institution selected by the user.

- ☐ **Storage**

  API identifiers are being stored (access_token, item_id, link_token, request_id, account_ id, etc.).
  If using Transactions - storing each user's transactions.

- ☐ **Callbacks**

  Listening to the onSuccess callback for the public_token & item_id.
  Listening to the onExit and onEvent callbacks for error_type & error_code.
  Listening to the onEvent callback for exit_status or timestamp.

- ☐ **Webhooks**

  A URL for Webhooks is established.

- ☐ **Update mode**

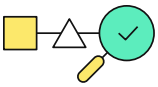
  To use update mode for an Item, initialize Link with a link_token configured with the access_token for the Item that you wish to update.

- ☐ **Removing Items**

  Items for churned users are deleted using /item/remove.
  Note: This is especially important if using recurring products such as Transactions, which will continue to be billed even if an Item is in an error state. Transactions is billed every month that an Item exists, and we recommend removing Items with recurring billing events near the end of the month if ongoing data is no longer needed.

- ☐ **Error handling**

  Calling /item/get to identify Items with an error status.
  Update mode is implemented to resolve errors (entry point included on UI).



Go live

Leverage the dashboard

- Institution status: The institution status dashboard reveals at-a-glance information about the health of each institution's integration. Use this or the API to help troubleshoot when a user reports difficulty connecting to a specific institution.
- Item status: Look up specific Items to see what institution that Item is connected to, when we last successfully updated or failed to update the Item (for Transactions), and information about webhooks associated with the Item. Use the Item status dashboard or API to dig deeper into the health of specific Items that your users are reporting issues about.
- Event logs: The event logs dashboard shows recent API requests and webhook activity. These logs can be filtered a number of different ways, including by error type or Item identifier.
- System status: The system status dashboard reveals at-a-glance information about the health of the Plaid API in each environment.

Support enablement

If you have a question or encounter an issue, first check out our docs, help center, and status page. If you still can't find the answer you're looking for, you can file a support ticket through the dashboard. Please be sure to select the right category for your ticket and to include all relevant identifiers and contextual information so our team can respond as quickly as possible.

Do you know how to check for Plaid's support of a particular institution?

- Our institutions endpoints allow you to check product support for every institution in our list. Keep in mind that institution information including names and IDs may change over time. In addition, product coverage may change as Plaid builds new integrations and/or institutions rebuild infrastructure.

Are you familiar with our status page?

- Our status page reports API uptime metrics as well as any major events affecting performance among major institutions. This is the best first place to check for any known issues with top banks.

Returning user experience

Build a faster Link flow for existing Plaid users

The returning user experience (RUX) simplifies onboarding for users who have already connected a financial account with Plaid. If your application verifies the user's phone number in your onboarding flow before the user enters Plaid Link, you may enable the returning user experience.

There are two RUX flows:

- Pre-matched RUX: This flow requires the user to only enter their password for a given financial institution.
- Pre-authenticated RUX: This flow requires the user to enter a one-time passcode that's sent to their mobile device.

Item management

Removing churned Items

Are you removing unneeded Items?

If you decide that you're unlikely to re-engage a given user, you can make an /item/remove request for the user's Item(s). This is especially important if using Transactions, which will continue to be billed even if an Item is in an error state. Because Transactions is billed every month that an Item exists, we recommend removing Items near the end of the month. Note that if a user returns to your platform at a later date, they will need to re-authenticate in Link to create a new Item.

Some common metrics used for determining when to remove an Item include:

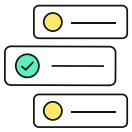
- Users who haven't opened the app / visited the site in 60 days.
- Users who haven't responded to an action in the last 10 notifications.
- Users who have been in an unresolved error state for more than 30 days.

Deduplication

Plaid won't prevent an end user from adding the same Item multiple times.

Use the metadata in the onSuccess() callback to compare against what you already have stored in your database before exchanging the public_token for an access_token.

- (e.g., institution, account name, account mask) to determine if they have previously linked their account(s).



Re-Link UI best practices

When a user changes their credentials, MFA, or security settings, or when a bank makes changes to its infrastructure, Plaid's connection to the user's bank account may be disconnected, which results in an ITEM_LOGIN_REQUIRED error.

When an Item enters an ITEM_LOGIN_REQUIRED state, the user's data is no longer retrievable. In order to resolve the error and begin accessing the user's data once again, you'll need to prompt the user to re-authenticate their account using Link's update mode. To successfully re-engage users, and to get them to re-authenticate their account, you can consider one or more of the following:

- Notify users in-app or via email if they have Items in an error state. Message why it's important to re-link their bank account, and ask the users to do so by taking them directly to the screen with Link's update mode.
- Alert users the next time they return to your application that there's a temporary problem with the connection to their bank account, which they can quickly fix by re-authenticating in Link's update mode.
- Notify users when the connection to their bank account has been re-established.

Conversion & reporting

There are many factors that go into optimizing Link conversion. Below are some guidelines and key areas to consider based off of what we've seen from some of our most successful clients:

Ensure you're tracking Link's onSuccess callback, onExit callback, and onEvent callback data, and that this data is accessible through your analytics platform of choice (Periscope, Looker, Tableau, Segment, or any comparable tool).

Define the steps across which you wish to assess conversion. For example, you might consider optimizing for the following two steps within Link:

- **Institution selection/Search → Credential pane:** In other words, what % of users find their bank and make it to the "Credential" pane?
- **Credential pane → Success:** In other words, of those that reach the "Credential" pane, what % successfully connect their account?

Other considerations

Assess the factors that may positively or negatively impact conversion. Generally speaking, conversion in Link may depend on:

- Industry/Use case
  - » Investment and lending services tend to have the highest conversion, followed by payments and personal finance management.
- Business model
  - » If users pay for your service, or the intent is that they will/may pay for it, conversion tends to be higher.
- Placement
  - » Where Link fits into the broader user experience can affect conversion rates. For example, by placing Link early in the onboarding flow: (1) You may gain more connected bank accounts which reduces downstream issues/ costs; (2) you can begin processing transaction data earlier (and deliver insights to the user faster); (3) the user may have less invested (in time, inputs, and understanding) in your application at that point (and therefore may be less likely to successfully convert).
- Plaid product use
  - » If you're initializing with Transactions (9,600+ institutions), you probably have a higher chance of a user finding his/her bank, but are also including a lot more smaller, less stable institutions. If you're initializing with Auth (2,300+ institutions), there's a higher chance the user can't find his/ her institution, but the success rate once they do tends to be higher. Regardless of product, it's likely the majority of your users will search for the top 25 institutions, so the impact noted here should be minimal.
- UI/UX
  - » The experience around Link, as well as how you customize the experience within Link, can have a tangible impact on Link conversion.

Glossary of common terms

TERM	DEFINITION	EXAMPLE USAGE
request_id	Unique identifier included in all Plaid success and error responses. The request_id identifies a response's corresponding API request and enables Plaid Support to identify requests in our logs.	"Please pass along the most recent **request_id** that corresponds to a request exhibiting the issue you've identified."
link_session_id	Included in the onExit, onEvent, and onSuccess callback of a Link integration.	"From the **link_session_id**, we can see that the user ended their Link session after selecting an institution in the 'Institution select' pane."
access_token	A rotatable token unique to a single Item; used in a customer's API request to access data for that Item. Including an access_token in support requests helps Plaid Support identify a user's Item.	"The user authenticated their credentials with Link, and an **access_token** was created successfully."
public_token	A short-lived token that can be exchanged for an access_token or used to initialize Link in update mode for an Item.	"A public_token is exchanged for an **access_token** in our exchange token flow."
account_id	Unique identifier used to differentiate an Item's accounts. An account_id is included in all successful responses for all Plaid products.	"I have confirmed that there are no transactions associated with the **account_id** flagged."
institution_id	Unique identifier used to differentiate banks and financial institutions supported by Plaid.	"We have upgraded our integration with Joe's Bank, which now operates under the **institution_id** is ins_123456."
transaction_id	Unique identifier used to differentiate an Item's transactions.	"I referenced the **transaction_id** you provided, and I've confirmed its amount matches what is on file with the financial institution."

Appendix

pending_transaction_id	Unique identifier used to map a posted transaction to its associated pending transaction when available.	“The transaction has since posted, and the **pending_transaction_id** parameter is now being populated.”
error_code	Included in an error response from Plaid’s API, an error_code identifies the specific error that an API request encountered.	“Our logs show that the **error_code** this user hit when attempting to add their Item is USER_SETUP_REQUIRED.”
client_id	Unique identifier for your Plaid account, which can be found under the “Keys” section of the Plaid dashboard. The client_id and secret are used in conjunction with an access_token to access data for an Item.	“We are seeing successful API traffic tied to your **client_id**.”
user_token	A user token generated using /user/create for Plaid’s Income product. Any Item created during the Link session will be associated with the user.	“We are calling the /credit/payroll_income/get endpoint using this **user_token**, but we are receiving an error back from the API.”
user_id	The Plaid user_id of the user associated with this webhook, warning, or error. Specific to Plaid’s Income product.	“I referenced the **user_id** you provided and have confirmed that the income data is ready to be retrieved.”
asset_report_token	A token that can be provided to endpoints such as /asset_report/get or /asset_report/pdf/get to fetch or update an asset report.	“I referenced the **asset_report_token** you provided and have confirmed that the report is ready to be retrieved.”
asset_report_id	A unique ID identifying an asset report.	“The **asset_report_id** provided pertains to the report that was available on xx date.”

Privacy & security

> s Plaid's privacy policy incorporated into your site's/app's privacy policy according to the MSA?

- Your Master Services Agreement (MSA) with Plaid requires you to ensure that each end user is put on notice of, and agrees to, Plaid's privacy policy located at https://plaid.com/legal.

Have you implemented best practices to prevent improper behavior, fraudulent or otherwise?

- It's important to layer in protective measures across your platform to prevent both unintentional misuse (e.g., a single user repeatedly unlinking/linking an account) and intentional misuse (e.g., a fraudster maliciously attempting to validate stolen bank credentials). Refer to the next section on best practices in more detail.

Security messaging: In addition to any security-related content within Link, and any relevant content with your privacy policy and other disclosures, there are a number of ways to describe your security posture and methods to end users. Some examples we have seen include:

- A standalone page devoted to security.
- A section within a broader "frequently asked questions" devoted to security: Text, iconography, and/or callouts about security around the Link flow.

If you choose to include some additional content around the security of Plaid as a trusted partner, we'd recommend checking out our security page as well as the suggested text below:

To authenticate accounts, we've integrated with Plaid, a San Francisco-based financial technology ("fintech") company that builds connections with thousands of financial institutions. They're a trusted partner to companies ranging from popular fintech applications to Fortune 500 companies and major banks. These relationships require the company to have strong security controls in place, and its platform is regularly audited and tested to ensure its controls meet industry standards. For example, Plaid ensures sensitive data (both inmotion and at-rest) is encrypted, and regularly completes SOC-2 Type 2 compliance, ISO27001 (Security) and ISO27701 (Privacy) audits. When you enter your online banking credentials, Plaid establishes an encrypted connection with your bank. This flow was designed with security in mind, as an easy alternative to entering static account and routing numbers. Once this connection is established, our application receives a token. This token enables us to get the data we need to provide our service without direct access to your bank account or credentials.

Actionable security best practices

The purpose of this document is to provide you, the Plaid developer customer, actionable best practices for protecting consumer data, and securing your applications and environments. What this document will highlight are common risk domains and industry standards for mitigating those risks. As a Plaid developer customer, you're expected to interpret these best practices in the context of your business, applications, and environments, as you're expected to have the most context and knowledge about your own business, applications, and environments. Additionally, Plaid has partnered and authored a fintech security best practices standard that can be accessed by visiting https://ofdss.org/.

OFDSS was created to help raise the bar for data security across the digital finance ecosystem while also continuing to foster innovation. It creates strong, auditable data security guidelines that maintain alignment with common and relevant criteria found in other security frameworks such as SSAE18 TSC for Security and NIST CSF, while providing clear requirements optimized for cloud-native, startups, and growth-stage companies.

Why does this matter?

As a Plaid developer customer, you will be interacting with consumer Personally Identifiable Information (PII) and providing a service directly to consumers. This means that maintaining consumer trust in security practices is critical to your success as a business. You can't establish and maintain this trust without deliberately identifying and mitigating security risks that are relevant to your business, applications, and environments.

Privacy vs. security:

The relationship between privacy and security is that privacy is an outcome that is enabled by security. In other words, a consumer's privacy can be compromised without security being compromised, but once security is compromised, so is the consumer's privacy. The following example scenarios further clarify this distinction:

- **Privacy breach:** A company intentionally sells consumer PII to a third-party for targeted advertising purposes that the consumer did not consent to. Notice that the consumer's privacy is now compromised, but there wasn't a data security breach.
- **Security breach:** A company experiences a security breach that exposes consumer PII and company intellectual property to attackers, and therefore, to the public. Notice that the consumer's privacy is now compromised as a result of this security breach.

Appendix

In the context of an application or an environment that collects, processes, or stores consumer PII, a security breach will always compromise consumer privacy, but it's possible to compromise consumer privacy without experiencing a security breach.

Since consumer privacy cannot be ensured without first identifying and mitigating security risks, this best practices guide will be security focused.

Security risks and mitigations:
Below is a framework for identifying risks and developing mitigations.

Identifying risks:
To identify security risks that are relevant to your business, applications, and environments:

1. Identify how your application and environment is interacting with sensitive data (non-exhaustive example below):

- Collection: You collect consumer transactions data through your Plaid integration.
- Processing: You add insights to the data using a data model.
- Storage: You store the transactions data and insights in a back-end database.

2. Identify the people, process, or technologies that are involved in the collection, processing, or storage of sensitive data (extending the non-exhaustive example below):

- Collection: You collect consumer transactions data through your Plaid integration.
  - People: No human involvement.
  - Process: Authenticated requests to the Plaid API.
  - Technology: Your back-end services and the Plaid API.
- Processing: You add insights to the data using a data model.
  - People: Data scientists maintain your data model.
  - Process: Data model analyzes and augments the data with insights.
  - Technology: Employee workstations and your data model.

- Storage: You store the transactions data and insights in a back-end database.
  - » People: No human involvement.
  - » Process: Data model writes insights and PII to your database.
  - » Technology: Your analytics model and your database instances.

3 **For each interaction (collection, processing, and storage) and people, processes, and technologies involved, determine what the underlying risks are to the sensitive data (further extending the non-exhaustive example below):**

- Collection: You collect consumer transactions data through your Plaid integration.
  - » People: No human involvement.
    - – Risk: N/A (no employees are involved in the collection of data).
  - » Process: Authenticated requests to the Plaid API.
    - – Risk: Unauthorized access to data in transit.
  - » Technology: Your back-end services and the Plaid API.
    - – Risk: Unauthorized access to your back-end services.
- Processing: You add insights to the data using a data model.
  - » People: Data scientists maintain your data model.
    - – Risk: Untested changes pushed to the data model.
  - » Process: Data model analyzes and augments the data with insights.
    - – Risk: Data model augmenting data with incorrect insights.
  - » Technology: Employee workstations and your data model.
    - – Risk: Malicious code executed on a workstation.

- Storage: You store the transactions data and insights in a back-end database.
  - People: No human involvement.
    - Risk: N/A (no employees are involved in the storage of data).
  - Process: Data model writes insights and PII to your database.
    - Risk: Database containing PII is compromised.
  - Technology: Your analytics model and your database instances.
    - Risk: Storage volumes containing PII are stolen.

In this example, we identified (1) major points of interaction with sensitive data, (2) the people, processes, and technologies that are involved in those points of interaction, and (3) some risks that each of those points of interaction pose to sensitive data. Next step is to identify and deploy mitigations for the identified risks.

Mitigating identified risks:

In the simple example highlighted in the identifying risks section, the collection, processing, and storage phases pose several risks to sensitive data. The goal now is to identify and deploy a control that mitigates each of those risks. Potential mitigations are highlighted below (extending the non-exhaustive example):

Collection: You collect consumer transactions data through your Plaid integration.

- People: No human involvement.
  - Risk: N/A (no employees are involved in the collection of data).

- Process: Authenticated requests to the Plaid API.
  - Risk: Unauthorized access to data in transit.
  - Mitigating control:
    - Transport Layer Security (TLS): All sensitive data in transit will use TLS 1.2 or better.

- Technology: Your back-end services and the Plaid API.
  - Risk: Unauthorized access to your back-end services.
  - Mitigating controls:
    - Strong authentication controls: Access to back-end services and hosting infrastructure will be gated by usernames, passwords, and a 2-factor authentication layer.
    - Access controls: Access to back-end services and hosting infrastructure will require approval, be frequently reviewed, and be revoked when no longer needed.

Processing: You add insights to the data using a data model.

- People: Data scientists maintain your data model.
  - Risk: Untested changes pushed to the data model.
  - Mitigating controls:
    - Change management: Changes to the data model will be tested prior to being deployed to production.
- Process: Data model analyzes and augments the data with insights.
  - Risk: Data model augmenting data with incorrect insights.
  - Mitigating controls:
    - Change management: Changes to the data model will be tested prior to being deployed to production.
    - Monitoring: Data model results will be monitored for accuracy.
- Technology: Employee workstations and your data model.
  - Risk: Malicious code executed on a workstation.
  - Mitigating control:
    - Endpoint security: All workstations will run up to date Host Intrusion Detection/Prevention (HIDS/P) agents (e.g., antivirus).

Storage: You store the transactions data and insights in a back-end database.

- People: No human involvement.
  - » Risk: N/A (no employees are involved in the storage of data).
- Process: Data model writes insights and PII to your database.
  - » Risk: Database containing PII is compromised.
  - » Mitigating controls:
    - – Object-level encryption: All sensitive data will be encrypted at object (or column) level before being written to the database.
    - – Key management: Master encryption keys will be stored separately from the encrypted data.
- Technology: Your analytics model and your database instances.
  - » Risk: Storage volumes containing PII are stolen.
  - » Mitigating controls:
    - – Sensitive data will only be stored using encrypted volumes.

At this stage of this simple hypothetical example, we have identified the controls that would mitigate the identified risks, and these mitigations should be deployed using what's considered "standard" for your underlying technology stack.

Appendix

Security control domains baseline:

A properly designed information security program will be reasonably exhaustive, and will make trade offs between certain risks and mitigations. However, certain risks are universal if you're collecting, processing, and storing sensitive data, and therefore, it is reasonable to expect certain control domains to be addressed in your information security program.

This section will highlight a minimum baseline of security control domains that are considered typical for any organization that might be collecting, processing, or storing sensitive data, and therefore, should be addressed in your information security policy. This table also provides potential mitigation controls as non-exhaustive or prescriptive examples.

CONTROL DOMAIN	RISK TO ADDRESS	POTENTIAL MITIGATING CONTROLS
Information security governance	1 Failure to define security objectives 2 Failure to allocate resources to security activities	• A documented and approved information security policy • Dedicated security functions and employees
Access management	3 Unauthorized access » Logical access » Physical access 4 Lack of segregation of duties 5 Weak authentication	• Requiring authorization (approval) for access to sensitive assets and data • Gating access to sensitive assets and data via strong authentication layers (username:password:2FA)
Assets management	6 Insecure acquisition 7 Insecure disposal 8 Lack of accountability 9 Lack of visibility into assets 10 Unpatched vulnerabilities » Misconfigurations » Outdated versions 11 Malicious code	• Defined procurement process for acquiring critical assets and assignment of ownership • Network MDM and asset discovery tools • Full-stack, recurring vulnerability scans • Patch management process • Host Intrusion Detection/Prevention Systems (HIDS/HIPS)

Change management	12 Unauthorized code changes 13 Untested code changes 14 Lack of version control 15 Lack of segregation of duties	• Defined build and release process and Continuous Integration/Deployment pipeline (e.g., DevOps CI/CD) • Unit, integration, and static code reviews prior to production deployments • Version controlled code repository (e.g., GitHub)
Cryptography	16 Failure to encrypt data » Data-in-transit » Data-at-rest 17 Failure to secure keys	• Transport Layer Security (TLS) for all HTTP requests throughout the stack • Envelope encryption of database objects if NoSQL DBs are used and cell-level encryption if SQL DBs are used • Encryption key storage and rotation using HSM or a cloud secrets manager (e.g., AWS KMS, Azure Key Vault)
Logging and monitoring	18 Lack of a reliable audit trail	• Logging material interactions with internal applications and services (e.g., AD GPO changes, code changes and deployments, changes to network ACLs) • Logging material interactions with your customer facing application (e.g., sign-up events, log-in attempts)
Network security	19 Failure to segment networks 20 Failure to prevent unauthorized network traffic » Ingress » Egress	• Segmenting production network into different subnetworks based on needs of assets (e.g., private, filtered, and public) • Use network ACLs to control inbound and outbound network traffic for subnetworks within the production network • Segmenting corporate network (e.g., one for employees and another for guests) • Use a Network Intrusion Detection/Prevention System (NIDS/NIPS) to inspect all network traffic • Filter inbound and outbound traffic between network endpoints using a DNS routing/selective proxying tool
Incident response	21 Failure to detect incidents 22 Failure to respond to incidents	• Defined process for responding to incidents (triage, investigation, mitigation, communication, post-mortem) • 24/7 rotating on-call schedules for responding to incidents

Vendor management	23 Failure to identify and mitigate vendor risks	• Defined vendor intake and diligence process and criteria for acceptance and rejection
Human resources	24 Failure to screen employees 25 Failure to train employees	• Background checks • Performance review cycles • Security awareness training
Independent testing	26 Lack of independence verification	• Using independent third-parties for network pen-testing, security-focused code reviews, and auditing • Crowdsourcing vulnerability detection through a public bug bounty program

Contextualizing your information security program:

Once you have identified the applicable risks, and deployed controls that mitigate those risks, you need a way to contextualize these risks and mitigations in an organized format in the form of an information security policy.

This will help you maintain and improve these controls as technology and threats change, and effectively communicate your information security posture to anybody that wants to understand the security risks that are relevant to your business, applications, and environment, and how you mitigate them.

To do this, every security control can be categorized by its objective and deployment layer. This is a universal framework for contextualizing an information security program. The objective of a security control can be preventive, detective, or corrective. The deployment layer for a security control can be physical, logical, or administrative. The following table captures this concept (extending the non-exhaustive example):

	PREVENTIVE	DETECTIVE	CORRECTIVE
PHYSICAL			
LOGICAL	• TLS • 2FA • Host IPS • Testing code changes • Object encryption • Volume encryption	• Host IDS • Monitoring	
ADMINISTRATIVE	• Access approval	• Access review	• Access revocation

Note: The empty spaces in this table are representative of the fact that the example used in this document is a non-exhaustive one. This is intentional for demonstrative purposes, your real information security controls should be able to populate all areas in this table. Once populated, you will be able to use this table to write your information security policy.

PLAID

Plaid powers the digital finance solutions that enable millions of people to live healthier financial lives. Trusted by 6,000+ of the world's leading companies and connected to 12,000+ financial institutions across the US, Canada, UK, and Europe, Plaid's mission is to unlock financial freedom for everyone.

plaid.com

PLAID HEADQUARTERS
1098 HARRISON ST.
SAN FRANCISCO, CA 94103