MARQETA

Marqeta Solutions Guide:
Impact Karma GPR: Karma Wallet

Introduction

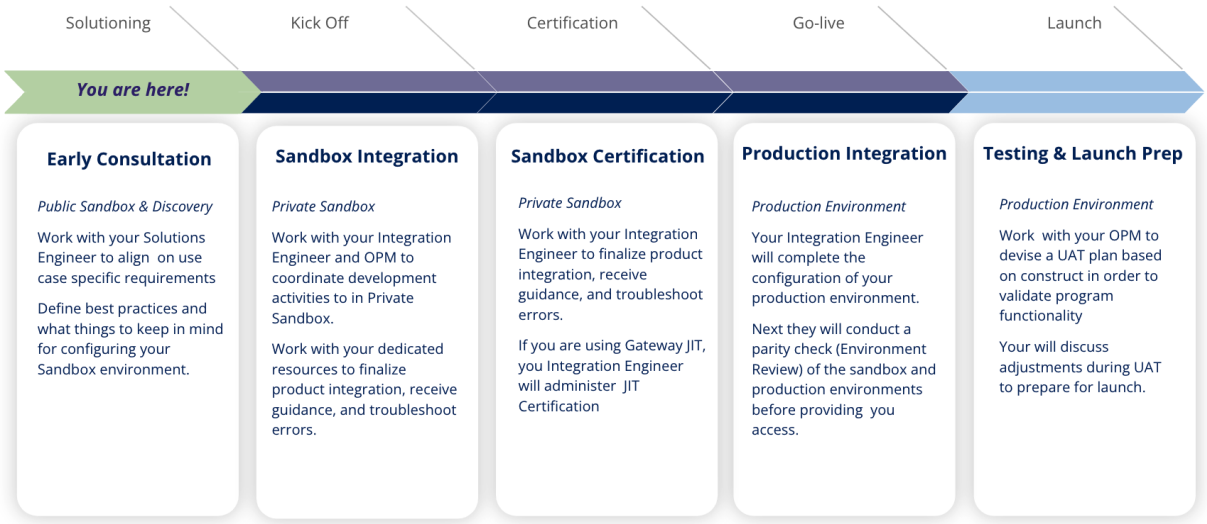
Objective

This Consumer GPR solution document will outline your technical journey during this “Managed By” integration with Marqeta. Your Sandbox and Production environments will be configured by Marqeta as your program manager. This document provides a detailed description on what Marqeta configures for you as well as a step-by-step guide on how to use our API to create users, cards and decisions on transactions.

Technical Journey

Here is an overview of the “technical journey” for your Marqeta integration.

After completing the steps outlined in this Solutions Guide you will be ready to work with our Integration Engineering team to prepare to move your solution into Production.


Solutioning
Kick Off
Certification
Go-live
Launch
You are here!
Early Consultation
Public Sandbox & Discovery
Work with your Solutions Engineer to align on use case specific requirements
Define best practices and what things to keep in mind for configuring your Sandbox environment.
Sandbox Integration
Private Sandbox
Work with your Integration Engineer and OPM to coordinate development activities to in Private Sandbox.
Work with your dedicated resources to finalize product integration, receive guidance, and troubleshoot errors.
Sandbox Certification
Private Sandbox
Work with your Integration Engineer to finalize product integration, receive guidance, and troubleshoot errors.
If you are using Gateway JIT, you Integration Engineer will administer JIT Certification
Production Integration
Production Environment
Your Integration Engineer will complete the configuration of your production environment.
Next they will conduct a parity check (Environment Review) of the sandbox and production environments before providing you access.
Testing & Launch Prep
Production Environment
Work with your OPM to devise a UAT plan based on construct in order to validate program functionality
Your will discuss adjustments during UAT to prepare for launch.

Getting Started

Pre-requisites

There is just one key technical dependency you will need to provide to Marqeta prior to receiving your Sandbox credentials:

- ☐ **Provide IP addresses for whitelisting.** Marqeta will need the IP addresses for any servers that will connect to the APIs in Sandbox and Production. Production environment IP addresses can be provided at a later date but Sandbox IP addresses are required before you will be able to access this environment.

Preparing for Development

Here is a list of the key activities through which you will progress as you begin your technical integration:

1. Familiarize yourself with the Developer Guides and detailed Core API specifications available on marqeta.com.
2. Set up a public sandbox account and begin experimenting with the Core API Explorer.
3. Review this Solutions Guide with a Marqeta Solutions Engineer
4. Provide IP addresses to Marqeta
5. Receive Sandbox environment credentials from Marqeta
6. Validate connection to Sandbox environment
7. Set up test data (user and card)
8. Submit test transactions
9. Develop, test and iterate on your Marqeta API integration

Technical Overview

Key Components of Technical Integration

Your technical integration with Marqeta's platform can be summarized into three main components:

Users, Accounts & Cards

Users, Accounts and Cards are the fundamental objects of interaction in your Marqeta program. You will integrate with Marqeta's APIs to:

- Manage the lifecycle of your cardholding users through creation, KYC (when applicable), activation, deactivation, etc.
- Manage the details and funding of Accounts and the relationships between Users, Cards, and Accounts
- Manage the lifecycle of cards through issuing, activation, suspending and termination

Ledger Management

Your system must incorporate a ledger management system to track the outcome of transactions and their impact on individual account balances. Marqeta sends transaction-level summary data using Webhooks. You will need to establish between one and five endpoints for consuming these Webhooks and incorporate ledger management capabilities as required for your program.

Design Principles for Your Marqeta Integration

Here's a short list of key principles to keep in mind with your integration:

- Marqeta's APIs are idempotent
- Manage events in chronological order by top level transaction token
- Be sure to respond to Webhooks once they are received. They will be re-sent until a successful response (HTTP 200) is acknowledged.

Resources

Integration Environments

Marqeta's customers typically interface with 3 distinct environment instances of our APIs - Public Sandbox, Private Sandbox, and Production.

- **Public Sandbox** is a multi-tenant shared environment that can be accessed by you at any time via our API Explorer. The sandbox is best used for understanding Marqeta APIs and sample requests/responses for various endpoints.
- **Private Sandbox** is configured by Marqeta and is a single tenant environment configured for the specific use case you will be utilizing in production. This should be considered as a Staging environment that will have the functional elements of the Production API except for the live production connection to the card networks and digital wallet providers.
- **Production** is configured by Marqeta and is a single tenant environment configured for your production launch according to your specific use case. Production will be connected to the card networks and will have a live enabled BIN associated with the issued cards.

Developer Documentation

Developer Guides: Cultivated guides for specific features and capabilities on the platform
Core API specifications*:* Detailed API specifications generated directly from the code base
Developer Community: Forums and content for developer education

Postman

Here's a Postman collection with a list of API calls unique to your business use case.

If you're creating requests in your private sandbox using Postman, please make sure the requests are coming from a whitelisted IP or configure your Postman to forward your requests through a proxy server.

Swimlanes

Visualized flow of the API configuration:
GPR

High Level API Calls

Creating the Account

1. POST/users
2. POST/kyc
3. POST/cards
4. POST/depositaccounts

Funding

1. POST/gpaorders
2. ACHR (originated outside the system with the account/routing number from the account)

Building & Testing

Users, Cards & Accounts

Marqeta will provide you with access to the Private Sandbox environment with the following configurations already in place in accords with the functional and regulatory requirements of your program:

Component	Description	Program-Specific Considerations
Card Products	Represent the card form factors (Virtual & Physical) and product configurations offered to your users	Virtual and Physical
Account Holder Groups	Represent a “class” of end users and the requirements for KYC/KYB.	KYC Required
Spend Controls	Program-level controls required by the Issuing Bank, including Velocity Controls, restricted countries, and restricted merchant categories	Set by the Bank

You will be responsible for managing the lifecycle of Users and Cards. To get started, here are some basic functions you will need to build into your integration:

KYC (Know Your Customer)

For your setup, KYC is a required piece of the User creation process. In order for a User to be submitted for KYC, certain fields must be populated. Those fields can be referenced in About KYC Verification.

API Endpoint	Commentary

POST/Users	Create a new User. Note that new User must be created using the DEFAULT_AHG account holder group token and must pass KYC before cards can be issued.
POST /kyc	Submit required fields for KYC

Users

Users are the cardholders to whom Cards will be assigned. Below are some of the basic operations you'll need to support. (Example API calls can be found in the Appendix section and in our Developer Documentation).

API Endpoint	Commentary
POST /users	Create a new User. Note that users will start in an ACTIVE state.
GET /users	Retrieve a list of Users
GET /users/{token}	Retrieve a specific User, referenced by its token
PUT /users/{token}	Update the attributes of a specific User.
POST /usertransitions	Change the state of a User (e.g. Suspend or Close)

User Management Tips:

- For the user token, we recommend specifying a unique identifier (UID) that is meaningful in your system. Once you have the token, store it for card creation. Note this cannot be changed once it is created.
- You are welcome to collect more or less information based on the key & value pairs that we offer outlined in our documentation.
- Note if you are using digital wallets (tokenization) or plan to in the future, you should include the user's email and phone number when creating the user since email & phone number is pulled from the user object to send a one time passcode to verify the user's identity during the step up process .
- You may also create any optional metadata in the /users endpoint by using the metadata object. This metadata will be included in usertransition Webhooks.

Cards

Card Products represent a “type” of card product in your program while Cards represent actual payment instruments that are assigned to and can be used by Users for payments.

API Endpoint	Commentary
GET /cardproducts	Retrieve a list of configured Card Products.
POST /cards	Create a card for a User
GET /cards/user/{token}	Retrieve cards associated with a User
POST /cardtransitions	Change the state of a Card (e.g. Suspend or Terminate)

PCI Considerations for PAN Access

Certain actions require access to sensitive card data that are in the scope of Payment Card Industry Data Security Standard (PCI DSS). All aspects of your program must adhere to these standards. Marqeta offers solutions both for PCI compliant and non-PCI compliant systems:

- **PCI Compliant:** Clients who are certified as PCI compliant have the option to access restricted API endpoints that allow full access to Primary Account Number (PAN) and Card Verification Value (CVV) for all cards as well as the ability to set PIN. You must provide proof of certification to receive this access.
- **Non-PCI Compliant:** Alternatively, Marqeta provides an embeddable Javascript library called Marqeta.js and customizable widgets that allow you to include necessary functionality into your application in a PCI-compliant manner without needing to obtain your own PCI certification. If you do not have PCI certification, you must use these components when accessing data in PCI scope.

Card Tips

- Use the metadata attribute to store transaction level or program-specific information you’d like to associate with the Card for reference in reporting or business logic.
- For **virtual cards**:
    - Virtual cards are “ACTIVE” upon creation
    - If your use case is approved to allow access to full PANs to the authorized consumer user, you will use the query parameters show_pan and show_cvv_number to present the virtual card details. The card fulfillment state will transition to DIGITALLY_PRESENTED. Please note, Marqeta will need to allow your API keys to see PCI details for this use case. (See: Displaying Card via Query Parameters)

    - If you intend to use Marqeta.js to present the virtual card to the end user, see: Using Marqeta.js
- For **physical cards:**
    - The card will be “UNACTIVATED” upon creation.
    - Other important things to include in the payload will be the shipping object along with the personalization object to include the user's name on the card.
    - We recommend listening to Card Transition Webhooks, “cardtransition.state” Webhook events for changes to the card state and “cardtransition.fulfillment” Webhook events for changes to the card fulfillment state changes.
    - Please note, Marqeta is required to conduct an OFAC check on users where the card is personalized with a name.
    - Use /cardtransitions to transition the card from “UNACTIVATED” to “ACTIVE” state at your discretion. This endpoint is only allowed if your use case is out of PCI scope. See: Using Activate Card and Set PIN Widgets

GPR Creation

API Endpoint	Commentary
POST /depositaccounts	For creating the Cardholder’s Marqeta Account TYPE=DEPOSIT_ACCOUNT A User or Business account must be created before a deposit account can be created. **An active card must also be created before the deposit account is created.**

After the creation of this resource- you will receive the account/routing number in the API response back which can be displayed to the end-user for setting up direct deposit etc. outside of the system.

Funds from the Impact Karma Program Funding Account

API Endpoints	Commentary
POST/gpaorders	Moves funds from the PFA to a Cardholder GPA.

POST/gpaorders/unloads	Returns funds to the PFA. Requires the token of the original load.

This allows Impact Karma to directly add funds to the user's GPR/Account for cashback, rewards etc.

Webhooks

About Webhooks

Webhooks are fired for a wide variety of Event Types that occur on the Marqeta platform. Webhook listeners are registered using the webhooks API. You can register between one and five active Webhook endpoints on your program.

Webhook listeners can be configured to receive all events ("*"), a subset of events of a particular type ("transaction.*") or specific events ("cardtransition.fulfillment.issued"). See About Webhooks for more information.

Your Integration Engineer will work with you to identify which Webhooks you will need to subscribe to depending on your program's specific requirements. When in doubt, it's better to subscribe to too many than not enough!

Response and Retries

- Your webhook listener is expected to respond with an HTTP 200 response to acknowledge that a Webhook has been successfully received. If there is no response or a failure response, the platform will continue to deliver the Webhooks at an exponentially increasing interval of 4^x seconds a maximum of ten times (4^{10} seconds ~ 12 days for the maximum interval). See Runtime Characteristics for more information.

- Webhooks are typically fired within seconds of the corresponding system event. Ledger updates based on Webhook events do not need to be "real-time" but should be considered "eventually consistent". Events will be re-sent at intervals of increasing duration until they are acknowledged with a HTTP 200 response.
- Webhooks are not guaranteed to be received in the order in which the triggering event occurs. Utilize the Webhook's created_time attribute and other contextual information contained in the Webhook message body when reconstructing a strict sequence of events is necessary

- Webhooks may be delivered in a batch, with multiple events received in a single message. Your Webhook listener should be able to receive and parse out multiple events within a single message.
- Ensure that your system of record is resilient enough to handle “exception” cases such as Webhooks that are delivered out of the expected order (e.g. a clearing is received before the originating authorization). These exceptions can occur for a wide variety of reasons outside the control of your system or the Marqeta platform.

Testing & Simulating Transactions

During the development lifecycle in the Sandbox environment, you can use Marqeta's Simulation API to test the behavior of the various resources configured into your Program environment. This is intended to ensure you have a strong grasp of the fundamental integration with Marqeta.

Simulate Transactions

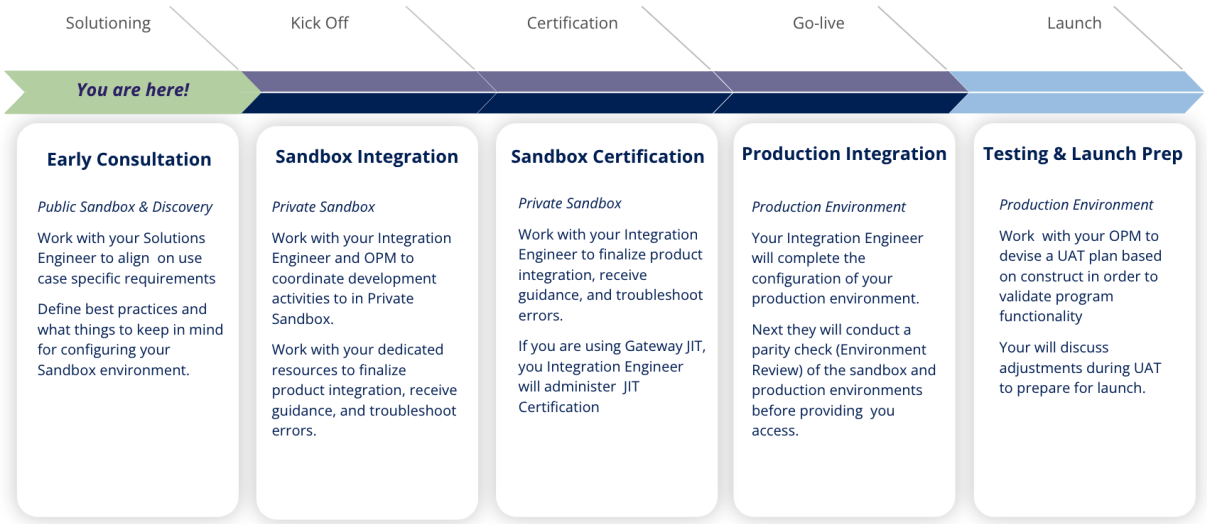
Marqeta provides the Simulate API as a convenient way to create simulated transactions that will generate Webhook messages that you can use to test your integration. Please note that the simulated transactions do not integrate with Card Networks or result in actual funds movement. This API is only available in Sandbox environments.

Here are a few of the most common transactions to simulate (see documentation for full list of transactions available for simulation):

API Endpoint	Commentary
POST /simulate/authorization	Simulate an authorization
POST /simulate/clearing	Clear a simulated transaction (can also be used for Returns)
POST /simulate/reversal	Reverse a simulated transaction
POST /simulate/authorization/advice	Simulate an advice adjustment to an existing simulated transaction

Preparing for Production

Timeline of Events



Once your sandbox integration is functionally complete and testable, you will work with your Solutions Engineer to determine readiness for the next stage of technical implementation with our Delivery Tech team.

Technical Readiness

In order to gain access to production you'll need to complete the following:

- Environment Review (1-2 weeks)

After our team is able to review this things, you will gain access to production API Keys.

Environment Review

Environment Review is when our team reviews your sandbox environment to ensure you've done it through testing. We make sure that your various objects are created correctly. Before environment review, you'll need to complete the following:

- Production IPs scanned and whitelisted
- Testing and Simulations Completed

Production Readiness

In order to receive your production API keys, you will need to have production IP addresses scanned for vulnerabilities and whitelisted. Note for sandbox, we don’t scan for vulnerability which is why turn around time is faster for whitelisting. Production IPs can take 3-5 business days since we scan those to check for vulnerabilities. Keep in mind that if we find vulnerabilities, you will need to fix them, this can delay whitelisting your production IP address.

Appendix

API Samples

API Call	Sample Call

POST /users	*Example includes all mandatory fields and a sample of optional fields:*
	
	{
	"first_name": "John",
	"last_name": "Smith",
	"token": "user626",
	"email": "john@gmail.com"
	"birth_date": "1991-01-01",
	"address1": "1234 Blake Street",
	"city": "Berkeley",
	"state": "CA",
	"country": "USA",
	"postal_code": "94702"
	}

POST /cards	*To issue a virtual card, insert the user's token in the payload. Use your card product token provided to you earlier.*
	
	{
	"token": "test_card_01",
	"card_product_token": "cardproduct_01",
	"user_token": "user626",
	}
	
	The card will be "ACTIVE" upon creation and ready for use.

Solution Stage Checklist

Defined below are task and milestones that mark progress toward your production implementation. Early progress toward these goals will ensure a faster and more seamless Production launch. .

- ☐ Review postman collection and technical assessment with Solution Engineer
  - ☐ Walk through onboarding process of user flow (web/mobile app signup)
  - ☐ Send Marqeta all of your source IPs for connecting to the private sandbox
- ☐ Get familiar with public sandbox and customized swimlanes
  - ☐ Good funds model of 6 day spend understood and allocated
  - ☐ Understand user to card creation using postman collection
- ☐ Review Marqeta suite of tools with Solutions Engineer
  - ☐ Marqeta Dashboard
  - ☐ Network Files
  - ☐ Reports (Diva)
- ☐ Begin ledger management dev work
  - ☐ Simulate transactions in private sandbox
  - ☐ Implement simple app using ngrok and simulate transactions to test JIT
  - ☐ Read and fully understand ledger management implementation
  - ☐ Draft ledger management database and app schema for JIT in private sandbox
  - ☐ Make a decision on whether to leverage Marqeta's STiP feature, Commando Mode.
- ☐ Refund Behavior
  - ☐ Read Online Refund Guide
  - ☐ Inform your solutions engineer if you plan to (1) Enable Online Refunds & (2) Use your GJIT system to review "pgfs.refund.authorization" messages (not enabled by default)
- ☐ AVS Behavior
  - ☐ Read AVS Guide
  - ☐ Inform your solutions engineer of your preferred AVS
  - ☐ Inform your solutions engineer if you choose to use your GJIT system to review "pgfs.authorization.account_verification" messages (not enabled by default)
- ☐ Gateway JIT dev setup for private sandbox
  - ☐ Set up private sandbox gateway endpoint
  - ☐ Send required information to solutions engineer
  - ☐ Simulate transactions in private sandbox
  - ☐ Begin testing ledger management for private sandbox
- ☐ MQ.js and activation widget
  - ☐ Brainstorm widget features using Marqeta Widget Style Preview page as a testing ground for you to determine how you wish for your widgets to look.
  - ☐ Design widget customization and provide details to your solutions engineer
- ☐ Final preparation for delivery kickoff
  - ☐ Review sandbox environment
  - ☐ Review transactions that will be sent in JIT Certification
  - ☐ Review hand-off expectations with Solutions Engineer