Informe Técnico Detallado: API de Bridge.xyz

1. Introducción a la API de Bridge.xyz

La API de Bridge.xyz se presenta como una solución integral para la orquestación y emisión de *stablecoins*. Su diseño busca ofrecer a los desarrolladores una experiencia ágil, comparable a la de plataformas como Stripe, para facilitar la conversión entre diversos formatos de dólar, incluyendo monedas fiduciarias (fiat) y *stablecoins* populares como USDC y USDP. Una capacidad distintiva de esta API es que permite a los desarrolladores no solo gestionar estas conversiones, sino también transformar estos activos en *stablecoins* personalizadas y programables, abriendo un abanico de posibilidades para la innovación en aplicaciones financieras.[1]

El propósito fundamental de la API es doble: por un lado, simplificar la complejidad inherente a la conversión entre diferentes representaciones del dólar y, por otro, empoderar a los desarrolladores con herramientas para la creación de sus propias *stablecoins* programables.[1] Esta capacidad de crear *stablecoins* personalizadas sugiere una plataforma orientada no solo a la facilitación de transacciones, sino también al fomento de nuevos casos de uso en finanzas descentralizadas (DeFi) o soluciones de pago a medida. La mención de la posibilidad de "ganar recompensas sobre activos" y que las "reservas se invierten en Bonos del Tesoro de EE. UU." [2], en el contexto de la emisión de USDB (la *stablecoin* de Bridge), apunta hacia un modelo de negocio que incentiva la emisión y utilización de *stablecoins* a través de su infraestructura.

La "orquestación" de *stablecoins* es un concepto central [1], indicando que Bridge.xyz se encarga de la coordinación y el manejo de las interacciones con múltiples rieles de pago, tanto fiduciarios como basados en blockchain. Esto reduce significativamente la carga de integración para los desarrolladores, quienes pueden interactuar con una única API para "moverse, almacenar y aceptar *stablecoins* sin problemas".[2] La plataforma ofrece un conjunto de productos que incluyen Orquestación, Emisión de *stablecoins*, Tarjetas, Carteras (*Wallets*) y Pagos Transfronterizos [2], delineando un ecosistema robusto para operaciones financieras.

Los casos de uso más comunes de la API abarcan desde rampas de entrada (*on-ramps*) y salida (*off-ramps*) para monedas fiat como USD, EUR y MXN, hasta intercambios entre diferentes *stablecoins* (por ejemplo, USDT por USDC). También facilita la creación de Cuentas Virtuales, que proveen a los clientes números de cuenta y ruta únicos para recibir USD y convertirlos a *stablecoins*, y Direcciones de Liquidación, que permiten crear direcciones de blockchain permanentes para recibir

depósitos cripto y liquidarlos automáticamente en fiat u otras *stablecoins*.[3]

1.1. Conceptos Clave de la API

La documentación de apidocs.bridge.xyz se estructura en torno a una serie de "Conceptos Clave de la API" [1], que representan los principales recursos y funcionalidades accesibles a través de su interfaz REST. La comprensión de estos conceptos es fundamental, ya que se referencian constantemente a lo largo de toda la documentación y son la base para una integración exitosa.[4]

La amplitud de estos conceptos clave, que se listan en la tabla a continuación, evidencia una API diseñada para ser exhaustiva, cubriendo el ciclo de vida completo de las transacciones fiat-cripto y la gestión de usuarios y sus activos. Desde la incorporación de usuarios (Customers, KYC Links) y la vinculación de métodos de pago (External Accounts), pasando por la ejecución de transacciones (Transfers, Liquidation Address, Virtual Accounts), hasta la notificación de eventos (Webhooks) y la monetización para el desarrollador (Developer Fees), la API aspira a cubrir un amplio espectro de necesidades en el ámbito FinTech.

Tabla 1: Conceptos Clave de la API de Bridge.xyz

Concepto Clave	Descripción Breve	Endpoint Principal Asociado (Inferido)
Autenticación (Authentication)	Cómo la API verifica la identidad del solicitante.[4]	N/A (Mecanismo de cabecera)
Idempotencia (Idempotence)	Cómo la API previene la ejecución duplicada de solicitudes.[4]	N/A (Mecanismo de cabecera)
Clientes (Customers)	Representación de los usuarios finales que utilizan los productos del desarrollador a través de Bridge.[4]	/v0/customers
Cuentas Externas (External Accounts)	Cuentas financieras de los usuarios (ej. bancarias) desde las cuales se pueden enviar o	/v0/customers/{id}/external_accounts

	recibir fondos.[4]	
Comisiones para Desarrolladores (Developer Fees)	Comisiones que los desarrolladores pueden establecer para cobrar a sus clientes por las transacciones.[4]	/v0/developer-fees [32]
Precisión y Redondeo (Precision and Rounding)	Manejo de la precisión numérica y el redondeo en operaciones financieras.[1]	N/A
Recibos (Receipts)	Detalles sobre cómo se entregaron los fondos salientes e información importante.[4]	Parte de la respuesta de Transferencias
Transferencias (Transfers)	Cómo iniciar el movimiento de dinero entre diferentes tipos de activos y rieles.[4]	/v0/transfers
Direcciones de Liquidación (Liquidation Address)	Una dirección de wallet permanente que envía los fondos depositados a un destino preconfigurado.[4]	/v0/customers/{id}/liquidation_addresses
Webhooks	Notificaciones asíncronas de eventos que ocurren en la plataforma Bridge.[4]	/v0/webhooks
Cuentas Virtuales (Virtual Accounts)	Un número de cuenta bancaria permanente que envía los fondos depositados a una dirección cripto preconfigurada.[4]	/v0/customers/{id}/virtual_accounts
Memos Estáticos (Static Memos)	Instrucciones de depósito fiat reutilizables para dirigir fondos a un destino cripto.[1]	/v0/customers/{id}/static_memos
Instrucciones de Depósito Fiat	Información necesaria para	Parte de respuestas de

	que los usuarios realicen depósitos fiat.[1]	Transferencias, etc.
Transacciones SEPA/Euro	Manejo de transferencias en la Zona Única de Pagos en Euros (SEPA).[4]	Parte de Transferencias, Cuentas Externas
Transacciones SPEI/MXN (Beta)	Manejo de transferencias a través del Sistema de Pagos Electrónicos Interbancarios (SPEI) de México.[1]	Parte de Transferencias, Cuentas Externas
Cuentas Prefinanciadas (Prefunded Accounts)	Cuentas que requieren fondos previos para operar.[1]	N/A
Tarjetas (Cards)	Emisión de tarjetas Visa para clientes en múltiples mercados.[4]	N/A (Endpoints específicos de tarjetas)

Fuente: [1]

2. Autenticación

La autenticación es el proceso mediante el cual la API de Bridge.xyz verifica la identidad del sistema o aplicación que realiza una solicitud. Es un paso fundamental y obligatorio para interactuar con la mayoría de los *endpoints* de la API.

2.1. Proceso de Obtención y Uso de Claves API

El mecanismo principal de autenticación para la API de apidocs.bridge.xyz se basa en el uso de claves API estáticas. El proceso para obtener y utilizar estas claves es el siguiente [5]:

1. **Creación de Cuenta:** Los desarrolladores deben primero registrarse y crear una cuenta de desarrollador gratuita en el panel de control de Bridge, accesible en dashboard.bridge.xyz. Un detalle particular de este panel es que utiliza un sistema de inicio de sesión sin contraseña, donde la autenticación se realiza a través de un enlace enviado al correo electrónico del desarrollador.
2. **Generación de Claves API:** Una vez dentro del panel de control, se debe navegar a la sección o pestaña denominada "API Keys". Allí, se puede generar una nueva clave API. Es crucial destacar que Bridge.xyz muestra la clave API

secreta *solo una vez* en el momento de su generación. Por lo tanto, es imperativo que el desarrollador copie esta clave de inmediato y la almacene en un lugar seguro y de acceso restringido.

3. **Uso de la Clave API:** Para autenticar las solicitudes a la API, la clave API generada debe incluirse en una cabecera HTTP específica: Api-Key. El valor de esta cabecera será la clave API secreta.

La clave API es un dato altamente sensible, ya que otorga acceso a las funcionalidades de la API en nombre de la cuenta del desarrollador. Por ello, se reitera la necesidad de mantenerla en un entorno seguro y de no exponerla en código del lado del cliente o en repositorios de control de versiones públicos.[5] En caso de que una clave API se vea comprometida, el panel de control de Bridge.xyz ofrece la funcionalidad para revocarla inmediatamente y generar una nueva, mitigando así el riesgo de acceso no autorizado.[5]

Si bien la documentación inicial referenciada en [1] no detallaba los métodos de autenticación y una búsqueda específica en la URL /docs/authentication no arrojó información, la guía de inicio rápido (/docs/quick-start) sí clarifica este proceso.[5] Es importante distinguir este método del descrito en [6], que se refiere a la API de docs.crunchybridge.com y menciona el uso de *Bearer tokens*, un sistema diferente al de apidocs.bridge.xyz.

La simplicidad de una clave API estática facilita una integración inicial rápida. Sin embargo, esta simplicidad también conlleva la responsabilidad de una gestión de claves extremadamente cuidadosa por parte del desarrollador. A diferencia de mecanismos más complejos como OAuth 2.0, que pueden ofrecer tokens de corta duración o capacidades de revocación más granulares, una clave API estática comprometida otorga acceso completo hasta que es revocada manualmente. La documentación en [5] no menciona explícitamente tokens de corta duración o mecanismos de rotación automática de claves para apidocs.bridge.xyz, lo cual es una consideración de seguridad importante.

2.2. Idempotencia

Para prevenir la ejecución duplicada de operaciones sensibles, especialmente aquellas que modifican datos (como las solicitudes POST), la API de Bridge.xyz soporta el concepto de idempotencia.[4] Esto se logra mediante el uso de una cabecera HTTP denominada Idempotency-Key.

Al realizar una solicitud que pueda tener efectos secundarios (por ejemplo, crear un

cliente o iniciar una transferencia), el desarrollador debe generar un valor único (como un UUID) y enviarlo en la cabecera Idempotency-Key.[5] Si la API recibe múltiples solicitudes con la misma Idempotency-Key dentro de un período de tiempo determinado, procesará la primera solicitud y devolverá la misma respuesta para las solicitudes subsecuentes, sin volver a ejecutar la operación.

La implementación de la idempotencia es una característica de diseño robusta que refleja una comprensión de los desafíos inherentes a la construcción de aplicaciones financieras fiables. Es crucial en sistemas donde las fallas de red o los reintentos automáticos podrían, de otro modo, llevar a la creación de múltiples registros no deseados o a la duplicación de transacciones financieras.[4] Se recomienda encarecidamente a los desarrolladores que implementen la generación y el envío de claves de idempotencia para todas las solicitudes de mutación (POST, y PATCH o PUT si fueran aplicables y soportadas con esta característica) para asegurar la resiliencia y la integridad de sus integraciones con la API de Bridge.xyz.

3. Gestión de Clientes (KYC/KYB)

La gestión de clientes, que incluye los procesos de Conozca a su Cliente (KYC) y Conozca a su Negocio (KYB), es un pilar fundamental de la API de Bridge.xyz, asegurando el cumplimiento normativo y la validación de la identidad de los usuarios finales.

3.1. API de Clientes (Customers API)

La API de Clientes (Customers API) permite a los desarrolladores integrar directamente los flujos de recopilación de información KYC/KYB en sus propias aplicaciones. Esto otorga un control independiente sobre la interfaz de usuario (UI) y la experiencia del cliente (UX), al tiempo que se delega el procesamiento y la validación de esta información a Bridge.[7]

Las **funciones principales** de esta API incluyen:

- Envío de información KYC para clientes individuales y KYB para clientes empresariales.
- Bridge.xyz monitorea el estado de verificación del cliente (kyc_status) y el estado de cada permiso o capacidad solicitada (denominados endorsements) para determinar si un cliente está autorizado a utilizar la plataforma y sus diferentes servicios. Por ejemplo, un endorsement base podría ser necesario para acceder a los rieles de pago de EE. UU..[7]
- Si la información proporcionada es válida y completa, el cliente recibe la

aprobación KYC y se le otorgan los endorsements solicitados.[7]
- En caso de que la información sea inválida, incompleta o inconsistente, el objeto del cliente devuelto por la API reflejará el estado KYC correspondiente, el estado del endorsement y los requisitos pendientes para ese endorsement.[7]

Para la **creación de clientes**, es un prerrequisito haber obtenido un signed_agreement_id. Este identificador se consigue después de que el usuario final acepta los Términos de Servicio de Bridge, un proceso que puede facilitarse mediante un enlace específico proporcionado por la API (como se detalla en [5]). Una vez obtenido, se puede proceder a crear el cliente mediante una solicitud POST al *endpoint* /v0/customers.[5]

Los **parámetros requeridos** varían significativamente entre clientes individuales y empresariales, reflejando la diferente naturaleza y complejidad de la información necesaria para cada tipo.

Tabla 2: Parámetros de Creación de Clientes (Individuales vs. Empresariales) - Selección Representativa

Parámetro	Descripción	Tipo de Dato	Ejemplo (Individual)	Ejemplo (Empresarial)	Requerido (Ind.)	Requerido (Emp.)
type	Tipo de cliente.	String	"individual"	"business"	Sí	Sí
first_name	Nombre del cliente.	String	"John"	N/A	Sí	N/A
last_name	Apellido del cliente.	String	"Doe"	N/A	Sí	N/A
email	Dirección de correo electrónico.	String	""	""	Sí	Sí
residential_address	Dirección residencial (para	Object	{ "street_line_1": "123	N/A	Sí	N/A

	individuos).		Main St", "city": "New York",... }			
registered_address	Dirección registrada (para empresas).	Object	N/A	{ "street_line_1": "456 Corp Ave", "city": "San Francisco",... }	N/A	Sí
birth_date	Fecha de nacimiento (YYYY-MM-DD).	String	"2007-01-01"	N/A	Sí	N/A
business_legal_name	Nombre legal completo de la empresa.	String	N/A	"My Business Inc."	N/A	Sí
business_type	Tipo de entidad empresarial.	String	N/A	"corporation"	N/A	Sí
signed_agreement_id	ID del acuerdo de Términos de Servicio firmado.	String	"d536a227-06d3-4de1-acd3-8b5131730480"	"7262662f-6622-46d1-9ac1-0e840d78598f"	Sí	Sí
identifying_information	Array de documentos de identificación (SSN,	Array	[{ "type": "ssn", "number": "xxx-xx-xxxx" }]	[{ "type": "ein", "number": "xxx-xx-xxxx" }]	Sí	Sí

	licencia, EIN).					
documents	Array de documentos de respaldo (estatutos, prueba de fondos, etc.).	Array	N/A (Puede ser requerido para EEE con proof_of_address)	[{ "purpose": "formation_document", "file": "base64_string" }]	No (general)	Sí (variable)
ultimate_beneficial_owners	Array de información sobre los beneficiarios finales reales (para empresas).	Array	N/A	``	N/A	Sí

Fuente: Basado en la estructura y ejemplos de.[7] La obligatoriedad puede variar según la jurisdicción y los endorsements solicitados.

Para **clientes internacionales o del Espacio Económico Europeo (EEE)**, pueden requerirse parámetros adicionales como número de teléfono, situación laboral, volumen esperado de pagos mensuales, y para clientes del EEE, documentos específicos como prueba de dirección (proof_of_address).[7]

La **respuesta de la API tras la creación o consulta de un cliente** incluye campos como el id del cliente, first_name (si aplica), kyc_status (que puede ser "not_started", "active", "under_review", o "rejected"), un array requirements_due que indica qué información adicional podría ser necesaria, y las marcas de tiempo created_at y updated_at.[7] Bridge.xyz revisa la información sometida y devuelve estados tanto para el KYC general (kyc_status) como para cada endorsement solicitado. Se recomienda utilizar ambos estados para determinar de manera integral la capacidad operativa del cliente en la plataforma.[7]

El **tiempo promedio de decisión para KYC** suele ser inferior a un minuto si la verificación es automatizada. Sin embargo, si se requiere una revisión manual (común

para algunas personas y todas las empresas), la decisión puede demorar hasta el siguiente día hábil. Las revisiones KYB para empresas suelen completarse el mismo día hábil o, a más tardar, el siguiente.[7]

La **progresión del estado KYC** sigue típicamente estos flujos [7]:

- not_started -> active: Es la ruta ideal para clientes individuales, donde la información pasa rápidamente las verificaciones automatizadas.
- not_started -> under_review -> approved o rejected: Este flujo aplica a todos los clientes empresariales y a algunos individuales que requieren una diligencia debida manual por parte del equipo de Cumplimiento de Bridge.
- not_started -> rejected: Puede ocurrir si información crítica, como el número de identificación fiscal, es incorrecta desde el inicio, impidiendo verificaciones posteriores.

La **actualización de la información de un cliente** se realiza mediante una solicitud PUT al *endpoint* /customers/:id. Es importante notar que, generalmente, se debe reenviar toda la información del cliente en la solicitud de actualización, incluso los campos que no han cambiado. Las actualizaciones parciales (enviando solo los campos modificados) están soportadas únicamente para un conjunto específico de parámetros como endorsements, residential_address, physical_address, verified_proof_of_address_at, y documents con el propósito proof_of_address.[7] Actualizar el objeto del cliente no borra los motivos de rechazo de intentos anteriores, pero tampoco impide una futura aprobación si se subsanan las deficiencias.[7]

La API de Clientes es, por tanto, crucial para la incorporación de usuarios y el cumplimiento de las normativas. La granularidad de los datos solicitados, especialmente para las entidades empresariales (como se observa en los numerosos campos para business en [7]), refleja los rigurosos requisitos de KYC y KYB en el sector FinTech. El hecho de que Bridge.xyz asuma una parte significativa de la carga de cumplimiento normativo [7] representa un atractivo considerable para las empresas que buscan integrar estas funcionalidades sin incurrir en la complejidad y el costo de desarrollar y mantener estos sistemas internamente. La dependencia de los endorsements [7] sugiere un sistema de permisos detallado donde diferentes niveles de verificación o tipos de cliente desbloquean distintas funcionalidades o acceso a rieles de pago específicos. La inaccesibilidad de la documentación específica sobre Endorsements [8] constituye un vacío informativo para comprender a cabalidad este mecanismo.

3.2. Enlaces KYC para Nuevos Clientes (KYC Links)

Como alternativa a la integración directa con la API de Clientes, Bridge.xyz ofrece los "KYC Links".[9] Esta funcionalidad está diseñada para simplificar el proceso de recopilación de información KYC/KYB, especialmente para aquellos desarrolladores que no desean o no necesitan construir una interfaz de usuario propia para este fin. Soporta tanto a clientes individuales como empresariales, tanto en Estados Unidos como a nivel internacional.[9]

Para **crear un Enlace KYC**, se realiza una solicitud POST al *endpoint* /v0/kyc_links, proporcionando el full_name (nombre completo del individuo o identificador único; para empresas, el nombre legal completo), email y type (tipo de cliente: individual o business). Opcionalmente, se puede incluir un redirect_uri para redirigir al cliente a una URL específica una vez completado el flujo, y un array de endorsements (por ejemplo, especificando sepa si se requiere una prueba de dirección para clientes no pertenecientes al EEE).[9]

La **respuesta de la API** a esta solicitud incluye dos enlaces cruciales [9]:

- tos_link: Un enlace que dirige al cliente a los Términos de Servicio de Bridge, los cuales deben ser aceptados antes de la aprobación.
- kyc_link: Un enlace que conduce a un flujo de KYC/KYB alojado y gestionado por un socio de Bridge, donde el cliente final somete su información y documentos para verificación.

El estado de un Enlace KYC puede ser verificado mediante una solicitud GET a /v0/kyc_links/:kyc_links_id, que devolverá el kyc_status, tos_status y, si el proceso fue aprobado, el customer_id asociado.[9] Los Enlaces KYC también pueden ser incrustados como un iframe dentro de la aplicación del desarrollador, para lo cual se deben pasar parámetros específicos en la URL del iframe (iframe-origin y modificar la sub-ruta de /verify a /widget).[9] Los flujos de estado y los tiempos de procesamiento son similares a los de la API de Clientes directa.[9]

Esta opción de Enlaces KYC demuestra la flexibilidad en el enfoque de Bridge.xyz hacia la integración de desarrolladores, atendiendo tanto a aquellos que buscan un control total sobre la experiencia del usuario (mediante la API de Clientes) como a aquellos que prefieren una solución más rápida y lista para usar. Es importante considerar que el uso de un socio externo para el flujo KYC/B [9] implica que los datos del cliente son manejados por un tercero durante este proceso, lo cual es una consideración relevante en términos de privacidad y flujo de datos.

3.3. Requisitos Adicionales para Clientes Existentes

En ciertas circunstancias, Bridge.xyz puede necesitar información adicional de clientes que ya han sido incorporados. Esto puede ocurrir, por ejemplo, para aumentar los límites de transacción (como en el caso de clientes de EE. UU. inicialmente restringidos que no proveyeron un ID gubernamental), o para habilitar el acceso a nuevos productos o servicios, como las transacciones SEPA/Euro.[10]

Para gestionar estos escenarios, la API proporciona mecanismos para solicitar esta información adicional:

- **Enlaces KYC para Clientes Existentes:** Se puede generar un enlace KYC específico para un cliente existente mediante una solicitud GET a /v0/customers/{customer_id}/kyc_link. Si el objetivo es habilitar servicios SEPA/Euro, se puede añadir el parámetro endorsement=sepa a la solicitud para indicar que se requiere una prueba de dirección.[10]
- **Aceptación de Nuevas Versiones de Términos de Servicio (ToS):** El acceso a ciertos servicios, como los pagos SEPA/Euro, puede requerir la aceptación de una versión actualizada de los ToS de Bridge. Para un cliente existente, se puede obtener un enlace de aceptación de ToS mediante una solicitud GET a /v0/customers/{customer_id}/tos_acceptance_link.[10] Este enlace puede ser presentado al cliente para su aceptación, y se puede configurar un redirect_uri para devolver al usuario a la aplicación del desarrollador con un signed_agreement_id como parámetro.

Estos mecanismos subrayan la naturaleza dinámica del cumplimiento normativo en el sector FinTech y cómo la API de Bridge.xyz está diseñada para adaptarse a estos cambios a lo largo del ciclo de vida del cliente, asegurando que tanto Bridge como sus clientes desarrolladores puedan cumplir con los requisitos regulatorios y de producto en evolución.[10]

3.4. Motivos de Rechazo de KYC (Rejection Reasons)

Cuando el proceso de verificación de un cliente resulta en un estado de kyc_status igual a rejected, la API de Bridge.xyz proporciona información sobre la causa del rechazo a través de dos campos específicos en el objeto del cliente [7]:

- developer_reason: Este campo está destinado exclusivamente para uso interno por parte de los desarrolladores que integran la API. Puede contener información técnica detallada o sensible que ayude a diagnosticar problemas con la información enviada o a identificar posibles intentos de abuso. No debe mostrarse directamente al cliente final.
- reason: Este campo ofrece una explicación más general y amigable para el

usuario sobre el motivo del rechazo. Está diseñado para que el desarrollador pueda comunicarlo directamente a su cliente final, ayudándole a entender qué falló y, potencialmente, cómo corregirlo.

Esta distinción es una práctica recomendada, ya que equilibra la necesidad de transparencia hacia el cliente con la provisión de información técnica valiosa para el equipo de desarrollo.

Tabla 3: Ejemplos de Motivos de Rechazo de KYC y su Mapeo

developer_reason (Motivo para el Desarrollador)	**reason (Motivo para el Cliente)**
"ID cannot be verified against third-party databases"	"Your information could not be verified"
"Inconsistent or incomplete information."	"Inconsistent or incomplete information."
"Cannot validate user age"	"Your information could not be verified"
"Missing or incomplete barcode on the ID."	"Cannot validate ID -- upload a clear photo of the full ID"
"Submission is blurry."	"Cannot validate ID - upload photo of ID is clear"
"ID is expired."	"ID is expired."
"No government ID found in submission."	"No government ID found in submission."
"PO box address detected."	"PO box address detected."
"Unsupported country"	"Your region is not supported"
"Missing or invalid proof of address"	"Missing or invalid proof of address"

Fuente: [11]

Comprender estos motivos de rechazo permite a los desarrolladores construir flujos de incorporación más robustos, proporcionar retroalimentación útil a los usuarios y

mejorar las tasas de aprobación de KYC/KYB.

4. Cuentas Externas (External Accounts)

Las Cuentas Externas en la API de Bridge.xyz representan las cuentas financieras propiedad de los usuarios finales, como cuentas bancarias o, potencialmente, tarjetas de débito, que se utilizan principalmente para retirar fondos (realizar *off-ramps* desde *stablecoins* a fiat).[12] Una ventaja significativa es que Bridge.xyz se encarga de la validación de estas cuentas externas y de los procesos KYC/KYB asociados al usuario, liberando al desarrollador de estas complejas responsabilidades de cumplimiento.[12]

Existen dos métodos principales para agregar Cuentas Externas [12]:

1. A través de la API de Bridge Directamente:
   Los desarrolladores pueden agregar cuentas externas mediante una solicitud POST al endpoint /v0/customers/{customer_id}/external_accounts.
   - **Parámetros de la Solicitud:**
     - type: Un campo legado que siempre debe enviarse con el valor "raw".
     - bank_name: Nombre de la entidad bancaria (ej. "Chase").
     - account_number: Número de cuenta bancaria del usuario.
     - routing_number: Número de ruta del banco.
     - account_owner_name: Nombre del titular o beneficiario de la cuenta.
     - address: Un objeto que detalla la dirección del titular de la cuenta, incluyendo street_line_1, city, state, postal_code, y country. Este campo es obligatorio.
     - account_name (opcional): Un nombre o alias para la cuenta (ej. "Cuenta de Ahorros").
     - active (opcional): Un booleano para indicar si la cuenta está activa.
   - **Respuesta de la API:** Una creación exitosa devuelve un objeto JSON con los detalles de la cuenta externa creada, incluyendo su id único, el customer_id al que pertenece, bank_name, account_owner_name, los últimos cuatro dígitos del número de cuenta (last_4), y las fechas de creación (created_at) y última actualización (updated_at).
2. A través de Plaid:
   Bridge.xyz ofrece una integración con Plaid Link, una solución popular para vincular cuentas bancarias de forma segura.
   - **Flujo de Integración con Plaid:**
     1. **Solicitar un link_token de Plaid a Bridge:** La aplicación del desarrollador realiza una solicitud POST a /v0/customers/{customer_id}/plaid_link_requests. La respuesta de Bridge

incluye el link_token necesario para inicializar el SDK de Plaid Link, la fecha de expiración de dicho token (link_token_expires_at), y una callback_url de Bridge.

2. **Iniciar el SDK de Plaid Link:** La aplicación del desarrollador utiliza el link_token para presentar la interfaz de Plaid Link al usuario final. Es crucial configurar la devolución de llamada (onSuccess) del SDK de Plaid para que apunte a la callback_url proporcionada por Bridge. Se recomienda que la llamada para obtener el link_token se realice desde el servidor del desarrollador para proteger la clave API de Bridge.
3. **Enviar el public_token de Plaid a Bridge:** Una vez que el usuario vincula exitosamente su cuenta a través de Plaid Link, el SDK de Plaid devuelve un public_token a la aplicación del desarrollador. Este public_token debe ser enviado a Bridge mediante una solicitud POST a la callback_url recibida en el primer paso (ej. /v0/plaid_exchange_public_token/{uuid_del_link_token}).
4. **Creación Asíncrona de Cuentas:** Tras el intercambio exitoso del public_token, Bridge recupera de forma asíncrona la información de las cuentas vinculadas desde Plaid y crea, de manera idempotente, los correspondientes objetos de Cuenta Externa en su sistema. Este proceso puede tomar unos minutos.

- **Consideraciones Importantes para Plaid:**
  - Algunos bancos están eliminando gradualmente el soporte para la vinculación de cuentas a través de WebViews; se deben consultar las recomendaciones de Plaid.
  - Plaid actualmente no soporta transferencias *wire* para ciertas instituciones como Chase y Bank of America. Para habilitar *wires* con estos bancos, la cuenta externa debe crearse utilizando el método de API directa de Bridge.

La disponibilidad de múltiples métodos para agregar cuentas externas ofrece flexibilidad. La integración con Plaid es una comodidad notable para los usuarios finales, aunque sus limitaciones, como el soporte variable para transferencias *wire* en ciertos bancos, pueden requerir el uso de la API directa de Bridge como una alternativa o complemento. La naturaleza asíncrona de la creación de cuentas después del intercambio del public_token de Plaid implica que las aplicaciones cliente deben implementar un mecanismo de sondeo o depender de webhooks (si Bridge los proporciona para este evento específico, lo cual no se detalla en [12]) para determinar cuándo las cuentas externas están disponibles y listas para ser utilizadas en transferencias. Aunque los documentos [13] y [13] sobre la API de Cuentas Externas eran

inaccesibles[12] proporciona una base sólida de información.

5. Transferencias (Transfers API)

La API de Transferencias es el componente central de Bridge.xyz para la movimentación de fondos. Está diseñada para ser una herramienta generalizada y flexible, permitiendo a los desarrolladores construir una variedad de flujos financieros que involucran tanto monedas fiduciarias como criptomonedas.[14]

5.1. Visión General de las Transferencias

Fundamentalmente, una transferencia en Bridge.xyz requiere la especificación de un source (origen de los fondos) y un destination (destino de los fondos).[14] Estos pueden ser una combinación de:

- Orígenes fiat (cuentas bancarias, tarjetas de débito, etc.) y destinos cripto (direcciones de *wallet* en diversas blockchains).
- Orígenes cripto y destinos fiat.
- Orígenes cripto y destinos cripto (por ejemplo, para intercambios entre diferentes *stablecoins* o movimientos entre blockchains).

La API soporta una amplia gama de monedas digitales y rieles de pago fiduciarios para facilitar estas conversiones.[14] Un aspecto importante del diseño es cómo maneja las transferencias donde Bridge no puede extraer fondos directamente del origen. Esto es común en escenarios como depósitos mediante transferencia bancaria (*wire*) o depósitos de criptomonedas, donde el usuario debe iniciar activamente el envío de fondos. En tales casos, la API de Bridge, en lugar de ejecutar la transferencia inmediatamente, devuelve un objeto source_deposit_instructions. Este objeto contiene la información necesaria (como detalles bancarios de Bridge o una dirección de depósito cripto) para que el cliente o su usuario final inicie la transferencia de fondos hacia Bridge.[14] Una vez que Bridge detecta la recepción de estos fondos, el resto del flujo de la transferencia (como la conversión y el envío al destino final) procede automáticamente sin necesidad de llamadas API adicionales por parte del desarrollador.[14] Este modelo de "empuje" para los fondos entrantes introduce un paso asíncrono en el flujo, que las aplicaciones integradoras deben gestionar adecuadamente en su interfaz y lógica, guiando al usuario sobre cómo completar el depósito requerido.

5.2. Transferencia de Fiat a Stablecoin (Crypto On-Ramp)

Este tipo de transferencia facilita la conversión de moneda fiduciaria tradicional

(como USD o EUR) en *stablecoins* (como USDC).[15] Soporta métodos de pago fiduciarios como transferencias ACH Push y transferencias Wire.[15]

Para iniciar una transferencia de fiat a *stablecoin*, se realiza una solicitud POST al *endpoint* https://api.bridge.xyz/v0/transfers. Los **parámetros clave** de esta solicitud incluyen [15]:

- amount: La cantidad de moneda fiduciaria a convertir.
- on_behalf_of: El ID del cliente de Bridge en cuyo nombre se realiza la transferencia.
- developer_fee (opcional): Una comisión que el desarrollador puede cobrar por la transacción.
- source: Un objeto que detalla el origen fiduciario:
  - payment_rail: El método de pago (ej. "wire", "ach_push", "sepa").
  - currency: La moneda fiduciaria (ej. "usd", "eur").
- destination: Un objeto que detalla el destino de la *stablecoin*:
  - payment_rail: La blockchain de destino (ej. "ethereum", "polygon").
  - currency: La *stablecoin* de destino (ej. "usdc").
  - to_address: La dirección de la *wallet* del destinatario.

La **respuesta de la API** a una solicitud exitosa incluye un id único para la transferencia, el estado inicial (state, típicamente "awaiting_funds", ya que Bridge espera la recepción de los fondos fiat), y, de manera crucial, el objeto source_deposit_instructions. Este objeto contiene [15]:

- amount: La cantidad exacta de fiat que el usuario debe depositar.
- currency: La moneda del depósito.
- deposit_message: Un mensaje o memo de depósito único (a menudo comenzando con "BRG").
- Detalles bancarios de Bridge (nombre del banco, número de cuenta, número de ruta, etc.) a donde el usuario debe enviar los fondos.

Es de **vital importancia** que el cliente final incluya el deposit_message único proporcionado en las source_deposit_instructions como el memo o mensaje de la transferencia Wire, o en la descripción de la transferencia ACH Push, al enviar los fondos. Este mensaje es único para cada transferencia y es esencial para que Bridge pueda conciliar y procesar correctamente la transacción.[15] La omisión o incorrección de este mensaje puede llevar a retrasos significativos o incluso a la imposibilidad de procesar el depósito. La aplicación que integra Bridge debe, por lo tanto, mostrar claramente estas instrucciones al usuario y enfatizar la importancia del deposit_message. La respuesta también incluye un objeto receipt con un desglose

financiero de la transacción.[15]

5.3. Transferencia de Stablecoin a Fiat (Crypto Off-Ramp)

Este flujo permite la conversión de *stablecoins* almacenadas en una blockchain a moneda fiduciaria, que luego se deposita en una cuenta bancaria externa del usuario.[16]

El proceso se inicia con una solicitud POST a https://api.bridge.xyz/v0/transfers. Los **parámetros clave** son [16]:

- amount: La cantidad de *stablecoin* a convertir.
- developer_fee (opcional): La comisión del desarrollador.
- on_behalf_of: El ID del cliente.
- source: Un objeto que detalla el origen de la *stablecoin*:
  - currency: La *stablecoin* (ej. "usdc").
  - payment_rail: La blockchain de origen (ej. "ethereum").
  - from_address: La dirección de la *wallet* del usuario desde donde se enviarán los fondos (aunque con "Transfer Features" esto puede ser flexible).
- destination: Un objeto que detalla el destino fiduciario:
  - currency: La moneda fiduciaria (ej. "usd").
  - payment_rail: El método de pago fiduciario (ej. "ach", "wire").
  - external_account_id: El ID de la Cuenta Externa del cliente (previamente registrada en Bridge) donde se depositarán los fondos fiat.

La **respuesta de la API** incluye el id de la transferencia, el estado inicial (state, usualmente "awaiting_funds"), y el objeto source_deposit_instructions. En este caso, las instrucciones de depósito indican [16]:

- payment_rail: La blockchain a la que se deben enviar las *stablecoins*.
- amount: La cantidad de *stablecoin* esperada.
- currency: La *stablecoin* a depositar.
- from_address: La dirección de origen del usuario (informativa).
- to_address: La dirección de la *wallet* de depósito de Bridge a la cual el usuario debe enviar sus *stablecoins*.
- blockchain_memo (si aplica): Para blockchains que requieren un memo para identificar depósitos (ej. Stellar, Tron), este campo será proporcionado.

Es **absolutamente crítico** que el usuario incluya el blockchain_memo en su transacción de depósito de criptomonedas si la blockchain de origen lo requiere. La omisión de este memo puede causar retrasos muy significativos en el procesamiento

del *off-ramp* o incluso la pérdida temporal de la capacidad de rastrear los fondos.[16] La aplicación integradora tiene la responsabilidad de mostrar claramente esta información al usuario.

Para **transferencias** *Wire* **salientes**, se puede especificar un mensaje para el beneficiario incluyendo un campo wire_message dentro del objeto destination en la solicitud de transferencia. Este mensaje tiene reglas de formato estrictas: un máximo de 3 líneas, y cada línea con un máximo de 35 caracteres.[16] Para **transferencias SEPA salientes**, se puede incluir una referencia (sepa_reference) en el objeto destination, con una longitud de 6 a 140 caracteres y un conjunto de caracteres permitido específico.[16]

5.4. Transferencia de Stablecoin a Stablecoin

Esta funcionalidad permite a los usuarios transferir una *stablecoin* de una blockchain a la misma *stablecoin* en una blockchain diferente (por ejemplo, USDC en Ethereum a USDC en Polygon). Bridge.xyz se encarga de manejar las conversiones y los puentes necesarios internamente.[17]

El proceso es similar al *off-ramp*: el usuario inicia la transferencia enviando la *stablecoin* de origen a una dirección de depósito proporcionada por Bridge.[17] La solicitud POST a https://api.bridge.xyz/v0/transfers incluye [17]:

- amount, developer_fee, on_behalf_of.
- source: Objeto con payment_rail (blockchain de origen), currency (*stablecoin*), from_address (*wallet* de origen del usuario).
- destination: Objeto con payment_rail (blockchain de destino), currency (la misma *stablecoin*), to_address (*wallet* de destino del usuario).

La **respuesta de la API** proporciona el id de la transferencia, el estado "awaiting_funds", y las source_deposit_instructions con la to_address de Bridge para el depósito de la *stablecoin* de origen y el blockchain_memo si es necesario.[17] La importancia del blockchain_memo es igualmente crítica aquí.

Esta capacidad aborda un caso de uso relevante en el ecosistema multi-cadena, permitiendo a los usuarios mover sus activos de manera fluida entre diferentes redes sin necesidad de interactuar directamente con puentes (*bridges*) de terceros, que a menudo pueden ser complejos o presentar riesgos de seguridad.

A continuación, se presenta una tabla comparativa de los tipos de transferencia:

Tabla 4: Comparativa de Tipos de Transferencia de Bridge.xyz

Tipo de Transferencia	Dirección del Flujo	Parámetros Clave de Origen	Parámetros Clave de Destino	Consideraciones Importantes
Fiat a Stablecoin (On-Ramp)	Fiat -> Bridge -> Stablecoin	source.payment_rail (wire, ach_push), source.currency (usd)	destination.payment_rail (ethereum), destination.currency (usdc), destination.to_address	El usuario debe enviar fiat a la cuenta de Bridge incluyendo el deposit_message único.
Stablecoin a Fiat (Off-Ramp)	Stablecoin -> Bridge -> Fiat	source.payment_rail (ethereum), source.currency (usdc), source.from_address	destination.payment_rail (ach), destination.currency (usd), destination.external_account_id	El usuario debe enviar *stablecoin* a la dirección de Bridge, incluyendo blockchain_memo si es necesario. Mensajes/referencias para Wire/SEPA.
Stablecoin a Stablecoin	Stablecoin (Red A) -> Bridge -> Stablecoin (Red B)	source.payment_rail (ethereum), source.currency (usdc), source.from_address	destination.payment_rail (polygon), destination.currency (usdc), destination.to_address	El usuario debe enviar *stablecoin* a la dirección de Bridge (Red A), incluyendo blockchain_memo si es necesario.

Fuente: Sintetizado de [15]

5.5. Características de Transferencia (Transfer Features)

La API de Transferencias de Bridge.xyz ofrece opciones de configuración flexibles, denominadas "Transfer Features", que permiten a los desarrolladores implementar patrones de transferencia comunes de manera más sencilla. Estas características se

habilitan por transferencia individual utilizando el campo features en la solicitud de creación de la transferencia.[18]

1. **Cantidad Flexible (flexible_amount):**
    - **Propósito:** Permite crear una Transferencia que aceptará y procesará cualquier cantidad de fondos que se envíen utilizando las source_deposit_instructions asociadas, sin necesidad de especificar un monto exacto al momento de la creación.[18]
    - **Comisión del Desarrollador:** Dado que el monto no se conoce de antemano, las transferencias con flexible_amount solo admiten una comisión de desarrollador basada en porcentaje (developer_fee_percent), no una comisión fija (developer_fee).[18]
    - **Plantillas Estáticas:** Esta característica se habilita automáticamente para las Plantillas Estáticas si no se proporciona una cantidad durante su creación.[18]
2. **Plantillas Estáticas (static_templates):**
    - **Propósito:** Permite a los desarrolladores crear una plantilla de Transferencia a través de la API. Esta plantilla genera instrucciones de depósito permanentes (memo de origen, mensaje de *wire*, referencia SEPA) que pueden ser utilizadas múltiples veces por los usuarios para enviar fondos a un destino preconfigurado.[18]
    - **Registros de Transferencia:** Cada vez que se reciben fondos que coinciden con la plantilla, Bridge crea un nuevo registro de Transferencia en la API. El id de la plantilla original permanece constante, mientras que cada instancia de transferencia generada a partir de ella tendrá un id único.[18]
    - **Cantidad Opcional:** Es opcional proporcionar una cantidad al crear una Plantilla Estática. Si no se especifica, la característica flexible_amount se habilita automáticamente.[18]
    - **Estado:** La plantilla inicial permanece en el estado awaiting_funds mientras esté activa. Las transferencias individuales generadas a partir de ella seguirán su propio ciclo de vida de estados.[18]
    - **Modificación y Listado:** Las plantillas pueden modificarse utilizando el *endpoint* de actualización de transferencias; los cambios se aplican a fondos futuros. Todas las instancias de transferencia creadas a partir de una plantilla específica se pueden listar utilizando el parámetro template_id en el *endpoint* de listado de transferencias.[18]
3. **Permitir Cualquier Dirección de Origen (allow_any_from_address):**
    - **Propósito:** Por defecto, al crear una transferencia con origen cripto (como un *off-ramp* o una transferencia cripto-a-cripto), Bridge requiere que se especifique la dirección de la *wallet* de envío (from_address) en el momento

de la creación. Esta característica aborda casos de uso donde la dirección de envío es desconocida de antemano, como cuando un consumidor envía fondos desde un *exchange* centralizado que utiliza direcciones de envío comunes o temporales.[18]

- **Comportamiento:** Si allow_any_from_address está habilitada, no es necesario especificar una from_address al crear la transferencia. Cualquier fondo enviado a las source_deposit_instructions se conciliará con la transferencia, independientemente de la dirección de origen real del envío en la blockchain.[18]

Estas "Transfer Features" demuestran una madurez en el diseño de la API, anticipando necesidades comunes de los desarrolladores más allá de las transferencias simples y únicas. Características como static_templates y flexible_amount son ideales para casos de uso como direcciones de depósito reutilizables o la aceptación de cantidades variables, comunes en plataformas de pago o *exchanges*. La característica allow_any_from_address resuelve un problema práctico significativo al recibir fondos de *exchanges* centralizados. La combinación de static_templates con webhooks (asumiendo que los webhooks notifican sobre nuevas instancias de transferencia) podría permitir la construcción de sistemas de procesamiento de pagos altamente automatizados.

5.6. Estados de las Transferencias

Las transferencias en Bridge.xyz pueden pasar por varios estados a lo largo de su ciclo de vida. Comprender estos estados es vital para rastrear el progreso de una transferencia, manejar excepciones y comunicar información precisa al usuario final.[14]

Tabla 5: Estados de Transferencia de Bridge.xyz y sus Significados

Estado	Descripción Detallada	Posibles Acciones Siguientes / Naturaleza
awaiting_funds	Bridge está esperando recibir los fondos del cliente antes de poder comenzar a procesar la transferencia. Este estado solo existe cuando Bridge espera un depósito del cliente (ej. depósitos cripto, *wires*, ACH *push*).	Espera de acción del usuario (depósito).

in_review	Estado temporal que rara vez se activa. Si ocurre, generalmente se resuelve automáticamente en segundos. Si no se puede confirmar la información en 24 horas, Bridge contactará al desarrollador.	Usualmente se resuelve solo; posible contacto de Bridge si persiste.
funds_received	Bridge ha recibido los fondos y está en proceso de moverlos en nombre del cliente.	Procesamiento interno por Bridge.
payment_submitted	Bridge ha enviado el pago y actualmente está esperando verificación (ej. en la blockchain, o confirmación del banco receptor). El tiempo de espera varía según el riel de pago.	Espera de confirmación externa.
payment_processed	La transferencia ha sido completada. Los fondos han sido enviados al destino especificado.	Estado terminal exitoso.
undeliverable	Bridge no pudo enviar los fondos al destino especificado (ej. número de cuenta/ruta inválido, activo no soportado por el destino).	Estado terminal fallido; requiere investigación.
returned	Bridge envió el pago, pero recibió una notificación de que no fue exitoso. Los fondos han sido devueltos a Bridge, y el proceso de reembolso al remitente original está en curso.	Fondos en proceso de ser devueltos al origen.
refunded	Los fondos para esta transferencia han sido devueltos al remitente original.	Estado terminal de reembolso.

canceled	La transferencia ha sido cancelada. (Las transferencias solo pueden cancelarse cuando están en el estado awaiting_funds).	Estado terminal de cancelación.
error	Hubo un problema que impidió a Bridge procesar esta transferencia. Puede requerir intervención manual para resolverse. Bridge contactará si no lo ha hecho ya.	Estado terminal fallido; requiere intervención.

Fuente: [14]

La granularidad de estos estados permite un seguimiento detallado y la capacidad de manejar diferentes escenarios y excepciones de manera más efectiva. Por ejemplo, la distinción entre payment_submitted (enviado pero no confirmado) y payment_processed (confirmado) es crucial para entender la finalización real de una transacción. Se recomienda encarecidamente a los desarrolladores que implementen manejadores para los estados terminales (como payment_processed, refunded, error, undeliverable) y que utilicen webhooks para recibir actualizaciones de estado en tiempo real, permitiendo una respuesta rápida y una comunicación proactiva con sus usuarios.

5.7. Obtención y Eliminación de Transferencias

La documentación proporcionada a través de los fragmentos de investigación no contenía información específica sobre cómo obtener (listar o consultar por ID) o eliminar registros de transferencias.[1] Esta es una funcionalidad básica que se esperaría en una API RESTful (por ejemplo, *endpoints* como GET /transfers para listar, GET /transfers/{id} para obtener una específica, y potencialmente DELETE /transfers/{id} para eliminar, aunque la eliminación podría estar restringida o ser menos común para registros financieros). La ausencia de esta información en los materiales disponibles representa una laguna informativa significativa para los desarrolladores.

6. Direcciones de Liquidación (Liquidation Addresses)

Las Direcciones de Liquidación de Bridge.xyz ofrecen una ruta de pago permanente y

automatizada que vincula una dirección de blockchain específica a un destino preconfigurado, que puede ser una cuenta bancaria (para liquidación en fiat) u otra dirección de blockchain (para liquidación en otra criptomoneda).[21]

El **propósito principal** de una Dirección de Liquidación es simplificar drásticamente el proceso de aceptar depósitos de criptomonedas y convertirlos automáticamente a fiat u otra cripto. Una ventaja clave es que, a diferencia de las transferencias directas que pueden requerir una coincidencia exacta de la dirección de origen (from_address) o del monto (amount) del depósito, las Direcciones de Liquidación son más flexibles. Esto las hace particularmente útiles en escenarios donde los clientes envían fondos desde *exchanges* centralizados (como Coinbase o Binance), ya que estos a menudo utilizan direcciones de envío comunes o cambian las direcciones de retiro, haciendo difícil predecir la from_address. También son útiles cuando sistemas intermediarios o comisiones de red (gas) pueden debitar pequeñas cantidades del monto enviado, resultando en depósitos con montos ligeramente diferentes a los esperados. En tales casos, una Dirección de Liquidación es la solución recomendada porque el remitente no necesita ser especificado de antemano para que Bridge gestione y liquide los fondos del cliente.[21]

Las **funcionalidades clave** de la API de Direcciones de Liquidación incluyen [21]:

- **Crear una Dirección de Liquidación con Destino Fiat:** Permite vincular una dirección de blockchain (ej. una dirección de Ethereum para recibir USDC) a una cuenta bancaria para pagos en fiat (ej. USD mediante wire o ACH, o EUR mediante sepa). Para destinos wire y sepa, los fondos se envían prontamente tras la recepción del depósito cripto. Para pagos ACH, las transacciones se encolan y procesan como parte de un lote diario (a la 1:00 pm EST/EDT según el documento).
- **Crear una Dirección de Liquidación con Destino Cripto:** Permite vincular una dirección de blockchain a otra dirección de blockchain, facilitando conversiones automáticas de *stablecoin*-a-*stablecoin* entre diferentes redes.
- **Establecer Comisiones de Desarrollador Personalizadas:** Se puede anular la comisión de desarrollador predeterminada para una Dirección de Liquidación específica durante su creación.
- **Obtener una Dirección de Liquidación Existente:** Permite recuperar los detalles de una dirección previamente creada.

Los **parámetros comunes para la creación** de una Dirección de Liquidación, ya sea con destino fiat o cripto, incluyen [21]:

- chain: La blockchain de origen desde la cual se enviará la criptomoneda (ej.

"ethereum", "stellar").
- currency: La criptomoneda que se recibirá en la dirección de liquidación (ej. "usdc", "usdt").
- destination_payment_rail: El método de pago o la blockchain para el destino final (ej. "wire", "sepa", "polygon").
- destination_currency: La moneda del destino final (ej. "usd", "eur", "usdc").

Para **destinos fiat**, se requiere external_account_id (el ID de la cuenta bancaria externa vinculada). Adicionalmente, se pueden especificar mensajes o referencias para el pago final: destination_wire_message para transferencias *wire* (con formato estricto: máximo 3 líneas, 35 caracteres/línea) o destination_sepa_reference para transferencias SEPA (formato estricto: 6-140 caracteres, conjunto de caracteres limitado).[21]

Para **destinos cripto**, se requiere destination_address (la dirección de la *wallet* de destino).[21]

Opcionalmente, se puede incluir custom_developer_fee_percent para establecer una comisión de desarrollador específica para esa dirección.[21]

La **respuesta de la API** tras una creación exitosa incluye el id de la Dirección de Liquidación, la chain y currency de origen, la address (esta es la dirección de blockchain real, generada por Bridge, a la cual los usuarios deben enviar sus depósitos cripto), los detalles del destino configurado, y, muy importante, un blockchain_memo si la chain de origen es una que lo requiere (como Stellar). Este blockchain_memo DEBE ser incluido por el depositante en su transacción de envío para evitar retrasos en la liquidación.[21]

Existen **montos mínimos de depósito** para que la liquidación se complete: para USDT, se requiere un mínimo de $20 USD; para todas las demás monedas, el mínimo es de $1 USD. Los depósitos por debajo de estos mínimos no serán procesados.[21]

Las Direcciones de Liquidación son una solución robusta para aceptar pagos en criptomonedas de forma continua y automatizar su conversión y liquidación. Representan una forma más avanzada y flexible de gestionar flujos de entrada de cripto (y su posterior conversión) en comparación con las Transferencias únicas, siendo especialmente adecuadas para escenarios comerciales, recepción de ingresos recurrentes en cripto, o cualquier caso donde el remitente o la cantidad exacta no se conocen de antemano. La funcionalidad de no requerir from_address o amount exactos es similar a las características allow_any_from_address y flexible_amount de

la API de Transferencias [18], pero aquí está integrada en el concepto de una dirección de depósito permanente. Aunque [21] ofrece una buena cobertura, la inaccesibilidad de documentos como [22] (sobre Drains, o drenajes/extracciones de fondos) y [23] (API de Direcciones de Liquidación, aunque [21] es muy informativo) podría significar que faltan detalles sobre cómo se gestionan o extraen los fondos acumulados en estas direcciones.

7. Cuentas Virtuales (Virtual Accounts)

Las Cuentas Virtuales de Bridge.xyz proporcionan a los clientes del desarrollador un conjunto único de número de cuenta y número de ruta (o equivalentes según la región, como IBAN para SEPA) que pueden aceptar transferencias bancarias tradicionales como *wires* y depósitos ACH.[24]

El **propósito fundamental** de las Cuentas Virtuales es actuar como una rampa de entrada (*on-ramp*) fluida desde monedas fiduciarias hacia *stablecoins* u otros activos en blockchain. Facilitan la recepción de fondos mediante métodos bancarios convencionales y luego enrutan automáticamente estos fondos a un destino de blockchain (una dirección de *wallet*) previamente especificado por el cliente del desarrollador.[24]

Las **funcionalidades clave** de la API de Cuentas Virtuales incluyen [24]:

- **Crear una Cuenta Virtual:** Se realiza mediante una solicitud POST a /v0/customers/{customer_id}/virtual_accounts. Se debe especificar la moneda de source (origen, ej. "usd") y los detalles de destination, incluyendo payment_rail (la blockchain de destino, ej. "ethereum", "stellar"), currency (la *stablecoin* o activo de destino, ej. "USDC", "USDT"), y la address (dirección de la *wallet* de destino). Se puede establecer un developer_fee_percent opcional (por defecto 0%). Para blockchains que utilizan memos (ej. Stellar), se puede incluir un blockchain_memo en el objeto destination. La respuesta de la API incluye el id de la cuenta virtual, su status (ej. "activated"), el customer_id asociado, las source_deposit_instructions (que contienen el nombre del beneficiario del banco, nombre del banco, dirección del banco, número de ruta, número de cuenta virtual único, y los rieles de pago soportados como ach_push y wire), y los detalles del destination configurado.
- **Listar Cuentas Virtuales:** Permite recuperar todas las cuentas virtuales asociadas a un customer_id específico.
- **Actualizar una Cuenta Virtual:** Permite modificar los detalles del destino (blockchain, moneda, dirección) o el developer_fee_percent de una cuenta virtual existente. Las transacciones futuras utilizarán los detalles actualizados. Esta

acción genera un evento account_update en el historial de la cuenta.

- **Desactivar una Cuenta Virtual:** Una cuenta desactivada no puede recibir nuevas transacciones; los fondos entrantes a una cuenta desactivada serán devueltos al remitente. Las transacciones que ya estaban en proceso antes de la desactivación se procesarán normalmente. Se dispara un evento deactivation.
- **Reactivar una Cuenta Virtual:** Permite que una cuenta previamente desactivada vuelva a procesar transacciones entrantes. Se dispara un evento reactivation.
- **Historial de Actividad:** Se puede obtener un historial de actividad para una cuenta virtual específica o para todas las cuentas virtuales de todos los clientes. Este historial incluye diversos **tipos de eventos**:
    - funds_received: Notificación de que Bridge ha recibido fondos y los está procesando.
    - payment_submitted: Bridge ha enviado el pago a la blockchain y está esperando verificación.
    - payment_processed: La transacción está completa y los fondos han llegado al destino en la blockchain (estado terminal).
    - funds_scheduled: Notificación de que los fondos entrantes se están procesando a través de un sistema externo con una fecha estimada de llegada (no para todos los pagos).
    - in_review: Estado temporal donde una transacción está bajo revisión; Bridge podría solicitar más información.
    - refunded: Se activa si Bridge no puede completar la transacción y devuelve los fondos al remitente.
    - microdeposit: Se activa cuando un banco emisor inicia un proceso de verificación por microdepósitos (estos fondos de prueba nunca se convierten a cripto).
    - account_update, deactivation, reactivation: Eventos no transaccionales relacionados con la gestión de la cuenta. Todos los eventos disparados por transacciones incluyen un deposit_id para vincular diferentes eventos (ej. funds_received, payment_submitted, payment_processed) a la misma transacción de depósito original. Los eventos funds_received proporcionan detalles sobre el depósito fiat. Los eventos payment_submitted y payment_processed informan sobre los fondos enviados *on-chain*, incluyendo el monto después de comisiones, el desglose de comisiones, y el destination_tx_hash (el hash de la transacción en la blockchain de destino, siempre poblado para payment_processed).

La **gestión** de Cuentas Virtuales se realiza mediante los *endpoints* REST estándar: POST para crear, GET para listar y obtener detalles, PUT para actualizar, y POST

específicos para desactivar/reactivar. Las **comisiones** para el desarrollador se establecen mediante developer_fee_percent, calculado como un porcentaje del monto de la transacción en la moneda de origen. Pueden aplicarse otras comisiones como exchange_fee_amount (por cambio de divisa si es necesario) y gas_fee (por la transacción *on-chain*).[24] El acceso a la funcionalidad de Cuentas Virtuales es un producto separado dentro de Bridge, con precios diferentes, y requiere contacto previo con Bridge para su habilitación.[24]

Las Cuentas Virtuales personalizan la experiencia de *on-ramp* fiat, ya que proporcionan a cada usuario final del cliente de Bridge (o a cada propósito específico) un número de cuenta bancaria "dedicado". Esto puede mejorar la confianza del usuario final y simplificar significativamente la conciliación de fondos para el cliente de Bridge, en comparación con un modelo donde todos los usuarios depositan en una cuenta bancaria centralizada de Bridge utilizando solo un memo para la diferenciación (como en el flujo de Transferencia Fiat-a-Stablecoin descrito en [15]). Son complementarias a las Direcciones de Liquidación (que manejan el flujo cripto-a-fiat/cripto [21]), cubriendo así los dos principales flujos de conversión de manera persistente y automatizada. Aunque el documento [25] sobre la API de Cuentas Virtuales era inaccesible, la información en [24] es exhaustiva.

8. Memos Estáticos (Static Memos)

Los Memos Estáticos en la API de Bridge.xyz son instrucciones de depósito de larga duración diseñadas para facilitar la aceptación de fondos fiduciarios (mediante transferencias *wire* y ACH *push*) y dirigirlos automáticamente a una *wallet* de destino preconfigurada en una blockchain.[26] Una característica notable es que el mismo Memo Estático puede utilizarse indistintamente para recibir tanto pagos *wire* como ACH *push*.[26]

Un aspecto **crítico** para el funcionamiento de los Memos Estáticos es que los clientes finales DEBEN incluir el deposit_message (un identificador único asociado al Memo Estático) cuando envían los fondos. La omisión o incorrección de este mensaje puede llevar a retrasos significativos en el procesamiento de los fondos o, en algunos casos, a la devolución de los mismos al remitente.[26]

Para **crear un Memo Estático**, se especifican los detalles del origen (source) y del destino (destination) [26]:

- **Origen (source):** Incluye la currency (ej. "usd") y el payment_rail (ej. "wire", aunque también soporta "ach_push" implícitamente).
- **Destino (destination):** Especifica el payment_rail de la blockchain (ej.

"ethereum", "stellar"), la currency de la *stablecoin* o activo (ej. "usdc", "usdt"), y la address de la *wallet* donde se enviarán los fondos convertidos. Para blockchains que requieren un memo (como Stellar), se puede incluir opcionalmente un blockchain_memo en el objeto destination.
- **Comisión del Desarrollador (developer_fee_percent):** Un campo opcional para establecer una tasa porcentual para las comisiones del desarrollador, que por defecto es 0 si no se especifica. Esta comisión se calcula sobre la moneda de origen, debe ser menor que el monto de la transacción del cliente y tiene un máximo de 5 dígitos de precisión.

La **respuesta de la API** tras una creación exitosa de un Memo Estático incluye [26]:

- id: Un identificador único para el Memo Estático (ej. "static_memo_alice_123").
- status: El estado actual, típicamente "activated".
- developer_fee_percent: La comisión del desarrollador configurada.
- customer_id: El ID del cliente de Bridge asociado con este memo.
- source_deposit_instructions: Información bancaria detallada para que los usuarios realicen los depósitos, incluyendo bank_name, bank_address, bank_beneficiary_name, bank_account_number, bank_routing_number, currency, y el crucial deposit_message.
- destination: Los detalles del destino configurado.

Los Memos Estáticos **no soportan** verificaciones por microdepósitos.[26] Existen **montos mínimos** requeridos para ciertas monedas de destino para completar el *on-ramp*: para USDT, se requiere un mínimo de $20 USD; para todas las demás monedas, el mínimo es de $1 USD.[26]

Otras **operaciones disponibles** para los Memos Estáticos incluyen listar todos los Memos Estáticos asociados a un cliente específico y actualizar un Memo Estático existente (modificando sus detalles de destino, lo cual también dispara un evento account_update en el historial del memo). También se puede acceder al **historial de actividad** de un Memo Estático, que es similar al de las Cuentas Virtuales e incluye eventos transaccionales (funds_received, payment_submitted, payment_processed, in_review, refund) y no transaccionales (account_update).[26]

Los Memos Estáticos parecen ser una versión quizás más simplificada o un precursor conceptual de las Cuentas Virtuales para el *on-ramp* de fiat a cripto. Ambos productos enfatizan la reutilización de instrucciones de depósito y la importancia crítica del deposit_message para la correcta atribución de los fondos. Una posible diferencia funcional radicaría en que las Cuentas Virtuales [24] proporcionan un número de cuenta y ruta *únicos por cuenta virtual creada*, mientras que los Memos Estáticos

podrían compartir detalles bancarios subyacentes de Bridge y diferenciarse principalmente por el deposit_message único asignado a cada Memo Estático. La funcionalidad de "Plantillas Estáticas" dentro de la API de Transferencias [18] también resuena con este concepto de instrucciones de depósito persistentes y reutilizables, sugiriendo un tema de diseño recurrente en la API de Bridge para facilitar los flujos de depósito.

9. Webhooks

Los Webhooks son un componente esencial de la API de Bridge.xyz, ya que permiten a las aplicaciones de los desarrolladores recibir notificaciones en tiempo real sobre eventos asíncronos que ocurren en la plataforma Bridge. Esto evita la necesidad de realizar sondeos (*polling*) constantes a la API para verificar cambios de estado, lo que resulta en arquitecturas de aplicación más eficientes y reactivas.

9.1. Configuración y Verificación de Firmas de Eventos

Para garantizar la autenticidad e integridad de las notificaciones de eventos enviadas a través de webhooks, Bridge.xyz implementa un mecanismo de verificación de firmas.[27] Cada vez que Bridge envía un evento a un *endpoint* de webhook configurado por el desarrollador, incluye una cabecera HTTP especial llamada X-Webhook-Signature. El formato del valor de esta cabecera es t=<timestamp_en_milisegundos>,v0=<firma_codificada_en_base64>.[27]

El proceso de **verificación de la firma** por parte de la aplicación receptora del webhook implica los siguientes pasos [27]:

1. **Extraer Componentes:** Parsear la cabecera X-Webhook-Signature para obtener el timestamp y la firma_codificada_en_base64.
2. **Construir el Mensaje Firmado:** Unir el valor del timestamp (como cadena) con el cuerpo (*body*) crudo de la solicitud HTTP recibida, utilizando un carácter de punto (.) como separador. Por ejemplo: <timestamp_valor>.<cuerpo_http_crudo>.
3. **Generar el Digest:** Calcular un *digest* SHA256 de la cadena construida en el paso anterior.
4. **Decodificar la Firma Recibida:** Realizar una decodificación en Base64 estricta de la firma_codificada_en_base64 extraída de la cabecera para obtener la firma binaria (decoded_signature).
5. **Verificar la Firma:** Utilizar la clave pública (public_key) asociada al *endpoint* del webhook (proporcionada por Bridge al momento de la creación del webhook), el digest calculado en el paso 3, y la decoded_signature obtenida en el paso 4 para realizar la verificación criptográfica. El algoritmo exacto (ej. RSA-SHA256,

ECDSA-SHA256) dependerá del tipo de clave pública, pero el principio es comparar la firma decodificada con una firma recalculada o verificar la firma con la clave pública y el digest.

Bridge aconseja que el *endpoint* receptor descarte eventos cuyo timestamp sea más antiguo que unos pocos minutos (por ejemplo, 10 minutos) respecto al tiempo actual del servidor receptor. Esto ayuda a prevenir ataques de repetición, donde un atacante podría interceptar un evento legítimo y reenviarlo posteriormente. Si un evento se considera demasiado antiguo, se recomienda devolver un código de estado HTTP 400 para solicitar a Bridge que reintente la entrega; Bridge generará un nuevo timestamp para cada intento de reenvío.[27] Se proporcionan ejemplos de código para la verificación de firmas en varios lenguajes de programación, incluyendo Ruby, Java, Python, JavaScript y Go.[27]

Los *endpoints* de Webhook se crean mediante una solicitud POST a /v0/webhooks. Inicialmente, se crean en un estado deshabilitado y deben ser habilitados explícitamente mediante una solicitud PUT para comenzar a recibir eventos. Existe un límite máximo de 5 webhooks (entre activos y deshabilitados) por cuenta. Es importante notar que, aunque se pueden crear *endpoints* de webhook en el entorno de Sandbox, no se enviarán eventos de webhook reales en dicho entorno.[28] La respuesta de la API a la creación de un webhook incluye su id, la url configurada, su status (activo, deshabilitado, eliminado), y la public_key en formato PEM, que es la que se debe utilizar para la verificación de firmas.[28]

El mecanismo de verificación de firmas de webhooks es robusto y sigue las mejores prácticas de la industria, utilizando criptografía de clave pública y *timestamps* para asegurar que las notificaciones provienen genuinamente de Bridge y no han sido manipuladas en tránsito. Es **imperativo** que los desarrolladores implementen correctamente la verificación de firmas en sus *endpoints* de webhook. Omitir este paso expondría su aplicación a riesgos de seguridad significativos, como el procesamiento de eventos falsificados.

A pesar de la claridad en *cómo* asegurar los webhooks, existe una laguna informativa crítica en los materiales proporcionados respecto a *qué* eventos específicos se envían y cuáles son sus estructuras de datos detalladas. Los documentos [29] (Visión General de Webhooks)[30] (Tipos de Eventos de Webhook)[31] y [31] (Estructura de Eventos) eran inaccesibles. Si bien se mencionan contextualmente algunos tipos de eventos en las secciones de Cuentas Virtuales [24] y Memos Estáticos [26] (como funds_received, payment_processed, account_update), una lista exhaustiva y las cargas útiles (*payloads*) de cada tipo de evento no están disponibles en los fragmentos. Esta

omisión es crítica para la implementación práctica y completa de la funcionalidad de webhooks.

10. Tarifas para Desarrolladores (Developer Fees)

La API de Bridge.xyz incluye una funcionalidad de "Tarifas para Desarrolladores" (Developer Fees) que permite a los desarrolladores que integran la API cobrar comisiones personalizadas a sus propios clientes por las transacciones procesadas a través de Bridge.[32]

Estas comisiones se deducen del monto de la transacción del cliente final. Bridge las mantiene en un libro mayor (*ledger*) separado, reservado para el desarrollador, y luego distribuye las comisiones acumuladas al desarrollador el día 5 de cada mes, presumiblemente a una cuenta externa previamente configurada por el desarrollador.[32]

Existen diferentes formas de aplicar estas comisiones según el producto de Bridge utilizado:

- **Cobro en Transferencias (Transfers):**
  - Para cobrar una comisión en una transferencia individual, el desarrollador debe establecer el parámetro developer_fee en la solicitud POST de creación de la transferencia.[32]
  - **Ejemplo:** Si el amount de una transacción es "$50.0" y el developer_fee se establece en "0.5", entonces $0.50 se reservarán para el desarrollador, y $49.50 (antes de otras comisiones de Bridge) se enviarán al destino especificado por el cliente.[32]
  - **Requisitos y Consideraciones Importantes para las Comisiones en Transferencias** [32]**:**
    - La comisión (developer_fee) se especifica siempre en USD y es un monto fijo, no un porcentaje.
    - Se establece por transacción individual.
    - La developer_fee debe ser siempre menor que el amount total de la transacción del cliente. Si es igual o mayor, la API devolverá un error.
    - La comisión puede tener un máximo de dos decimales (ej. 10.98 es válido, 10.987 no lo es).
    - El monto que finalmente recibe el destino será el amount original menos la developer_fee (y otras comisiones aplicables de Bridge).
    - Existen mínimos de transacción después de deducir la comisión del desarrollador: para transacciones con USDT, el monto neto debe ser de al menos $20; para todas las demás transacciones, debe ser mayor a $1.

        - El parámetro developer_fee es opcional; si se omite, por defecto es "0.0".
        - Esta comisión solo es aplicable a transacciones donde los fondos son transferidos *por* Bridge hacia un destino (ej. *off-ramps*, cripto-a-cripto), no cuando los fondos son transferidos *hacia* Bridge (ej. en la parte de depósito de un *on-ramp*).
- **Cobro en Direcciones de Liquidación (Liquidation Addresses):**
    - Los desarrolladores pueden establecer un porcentaje de comisión predeterminado (default_liquidation_address_fee_percent) que se aplicará a todas las Direcciones de Liquidación. Esto se configura a través de una API específica para la gestión de comisiones de desarrollador (referenciada como POST /developer-fees en [32]).
    - Adicionalmente, se puede establecer un custom_developer_fee_percent al momento de crear una Dirección de Liquidación específica, lo cual anulará el porcentaje predeterminado solo para esa dirección.[21]
    - **Ejemplo:** Si se depositan $50.00 en una dirección de liquidación con una comisión del 0.5%, $0.25 se agregarán a la cuenta de comisiones del desarrollador, y el cliente recibirá $49.75 (antes de otras comisiones de Bridge).[32]
- **Cobro en Cuentas Virtuales (Virtual Accounts) y Memos Estáticos (Static Memos):**
    - Tanto para Cuentas Virtuales [24] como para Memos Estáticos [26], se puede definir un developer_fee_percent. Esta comisión porcentual se calcula sobre el monto de la transacción en la moneda de origen.

Esta funcionalidad de Comisiones para Desarrolladores ofrece una vía directa para que los desarrolladores moneticen las aplicaciones y servicios que construyen sobre la infraestructura de Bridge.xyz. La flexibilidad para definir comisiones (fijas por transferencia, o porcentuales para productos de depósito recurrente como Direcciones de Liquidación, Cuentas Virtuales y Memos Estáticos) permite a los desarrolladores adaptar el modelo de ingresos a las características específicas de sus productos y a las expectativas de sus clientes. Los requisitos sobre el monto de la comisión (menor que la transacción, mínimos post-comisión) son validaciones importantes que los desarrolladores deben considerar al diseñar su estructura de precios.

11. Manejo de Errores

Un manejo de errores claro y predecible es fundamental para que los desarrolladores puedan construir integraciones robustas y resilientes con cualquier API.

11.1. Errores Generales y Códigos HTTP

La información específica sobre el manejo general de errores para la API de apidocs.bridge.xyz es notablemente escasa en los fragmentos de investigación proporcionados. El documento inicial [1] indica que la sección "getting-started" no ofrece una visión general de estos mecanismos, y una búsqueda en la URL /docs/handling-deposit-errors no produjo resultados. Adicionalmente, los documentos que parecen dedicados al manejo de errores, como [33] y [36], fueron reportados como inaccesibles.

Esta es una **laguna informativa crítica**. Sin una documentación oficial sobre la estructura de los mensajes de error en JSON, los códigos de error específicos de la aplicación (más allá de los de KYC) y las mejores prácticas para el manejo de errores de apidocs.bridge.xyz, los desarrolladores tendrían que recurrir a la experimentación o al soporte directo para comprender completamente cómo la API comunica las condiciones de error.

No obstante, es una práctica común que las API RESTful utilicen códigos de estado HTTP estándar para indicar el resultado general de una solicitud. Basándose en la documentación de APIs análogas (que debe tomarse solo como referencia general y **con extrema precaución, ya que podría no ser directamente aplicable a apidocs.bridge.xyz**), se podrían esperar los siguientes comportamientos [37]:

- **Códigos 2xx (ej. 200 OK, 201 Created):** Indican éxito.
- **Códigos 4xx (ej. 400 Bad Request, 401 Unauthorized, 403 Forbidden, 404 Not Found, 409 Conflict, 422 Unprocessable Entity, 429 Too Many Requests):** Indican un error del lado del cliente (ej. solicitud malformada, datos inválidos, falta de autenticación o autorización, recurso no encontrado, demasiadas solicitudes).
- **Códigos 5xx (ej. 500 Internal Server Error, 503 Service Unavailable):** Indican un error del lado del servidor de Bridge.

La API análoga docs.bridgeapi.io también devuelve un cuerpo de respuesta JSON para los errores, que típicamente incluye un código de error interno (code), el nombre de la propiedad o parámetro que causó el error (property), y un mensaje descriptivo legible por humanos (message).[37] Es posible que apidocs.bridge.xyz siga un patrón similar, pero esto no puede confirmarse con la información disponible.

Tabla 6: Códigos de Error HTTP Comunes (Referencia General – No Específico de Bridge.xyz)
Advertencia: La siguiente tabla se basa en convenciones HTTP estándar y ejemplos de otras APIs. Los detalles específicos, especialmente el cuerpo de la respuesta de error, para

apidocs.bridge.xyz pueden variar y no están documentados en los materiales proporcionados.

Código HTTP	**Mensaje HTTP Estándar**	**Posible Significado en el Contexto de una API**
200	OK	Solicitud exitosa.
201	Created	Recurso creado exitosamente (ej. un nuevo cliente, una nueva transferencia).
202	Accepted	La solicitud ha sido aceptada para procesamiento, pero este aún no ha finalizado (común en operaciones asíncronas).
204	No Content	Solicitud exitosa pero no hay contenido para devolver (ej. tras una eliminación exitosa).
400	Bad Request	La solicitud es malformada, sintaxis incorrecta, o contiene parámetros inválidos que no pasan una validación básica.
401	Unauthorized	La solicitud no pudo ser autenticada; falta la clave API, es incorrecta, o ha expirado/sido revocada.
403	Forbidden	La solicitud fue autenticada, pero el cliente no tiene los permisos necesarios para acceder al recurso o realizar la operación solicitada.
404	Not Found	El recurso solicitado no existe en el servidor (ej. un cliente con un ID inexistente, un *endpoint* incorrecto).

409	Conflict	La solicitud no pudo ser procesada debido a un conflicto con el estado actual del recurso (ej. intentar crear un recurso que ya existe con un ID único).
415	Unsupported Media Type	El servidor no puede procesar la entidad de la solicitud porque el formato de los datos no es compatible (ej. enviar XML cuando se espera JSON).
422	Unprocessable Entity	La solicitud está bien formada sintácticamente, pero contiene errores semánticos (ej. un valor de campo está fuera de rango, datos lógicamente inconsistentes).
429	Too Many Requests	El cliente ha excedido los límites de tasa de la API.
500	Internal Server Error	Ocurrió un error inesperado en el servidor de Bridge.
503	Service Unavailable	El servidor no está disponible temporalmente (ej. por mantenimiento o sobrecarga).

Fuente: Basado en [37] y convenciones HTTP estándar.

11.2. Errores Específicos de KYC

A diferencia de los errores generales, el manejo de errores para el proceso de KYC/KYB sí está documentado con más detalle. Como se explicó en la Sección 3.4 (Motivos de Rechazo de KYC), cuando la verificación de un cliente falla, la API devuelve los campos developer_reason (para diagnóstico interno del desarrollador) y reason (para comunicar al cliente final) en el objeto del cliente.[7]

12. Estructuras de Datos Principales

Al igual que con el manejo general de errores, la documentación formal y centralizada sobre las estructuras de datos (modelos o esquemas) para la API de apidocs.bridge.xyz no se encontró en los fragmentos de investigación. El documento [1] señala que la página "getting-started" no detalla explícitamente estas estructuras, y las búsquedas en la referencia de la API (/reference) [38] tampoco arrojaron un compendio de modelos de datos.

Se infiere lógicamente que cada concepto principal de la API (Clientes, Cuentas Externas, Transferencias, Direcciones de Liquidación, Cuentas Virtuales, etc.) tendrá sus propias estructuras de datos específicas tanto para las solicitudes (request payloads) como para las respuestas (response payloads). Sin embargo, sin esquemas formales como una especificación OpenAPI (anteriormente Swagger), que [5] menciona que los desarrolladores podrían generar por su cuenta pero que no se proporciona directamente en los materiales, los desarrolladores deben depender de:

- Los ejemplos de cuerpos de solicitud y respuesta JSON que se encuentran dispersos en la documentación de cada *endpoint* individual (por ejemplo[7] para Clientes[15] para Transferencias Fiat-a-Stablecoin[24] para Cuentas Virtuales[21] para Direcciones de Liquidación, etc.).
- La inferencia de tipos de datos y obligatoriedad de campos a partir de estos ejemplos.

Esta ausencia de una referencia de modelos de datos centralizada y formal puede presentar desafíos para los desarrolladores, especialmente en lo referente a la validación de datos del lado del cliente, la comprensión completa de todos los campos posibles (incluyendo los opcionales o condicionales), sus tipos de datos exactos y sus restricciones.

Para los propósitos de este informe, las estructuras de datos relevantes se han descrito contextualmente dentro de cada sección que detalla una funcionalidad específica de la API, basándose en los ejemplos proporcionados en los fragmentos. Por ejemplo, la Tabla 2 en la Sección 3.1 resume los parámetros para la creación de clientes.

13. Políticas de la API (Específicas para apidocs.bridge.xyz)

Las políticas de una API, como los límites de tasa y las estrategias de versionado, son cruciales para que los desarrolladores diseñen aplicaciones que interactúen de manera eficiente y sostenible con el servicio.

13.1. Límites de Tasa (Rate Limiting)

No se encontró información concreta sobre los límites de tasa específicos para la API de apidocs.bridge.xyz en los fragmentos de investigación proporcionados. Los documentos que podrían contener esta información ([39] sobre políticas de límites de tasa) fueron reportados como inaccesibles, y el documento sobre el entorno Sandbox [41] no menciona límites.

Si bien existen referencias a límites de tasa en la documentación de APIs con nombres similares (como docs.bridgeapi.io [37], que menciona límites como 2000 solicitudes por aplicación cada 5 minutos, y getbridge.com [43], que habla de límites dinámicos), esta información **no debe asumirse como aplicable** a apidocs.bridge.xyz, ya que pertenecen a productos o empresas potencialmente diferentes.

La ausencia de esta información es una deficiencia documental grave, ya que los límites de tasa son vitales para el diseño de aplicaciones de producción, para evitar ser bloqueado por la API y para asegurar un uso justo de los recursos. Los desarrolladores necesitarían obtener esta información directamente del soporte de Bridge.xyz o de documentación más actualizada.

13.2. Versionado y Changelog (Registro de Cambios)

De manera similar a los límites de tasa, la información detallada sobre la estrategia de versionado de la API de apidocs.bridge.xyz y la ubicación de un registro de cambios (*changelog*) es limitada o inexistente en los fragmentos proporcionados. Los documentos que podrían tratar estos temas ([44] para el *changelog*; [45] para versionado) fueron reportados como inaccesibles.

La única pista sobre el versionado proviene de la estructura de las URLs de los *endpoints* que se muestran en varios ejemplos a lo largo de la documentación (ej. [5]), las cuales consistentemente incluyen /v0/ en la ruta (ej. https://api.bridge.xyz/v0/customers). Esto implica fuertemente que la API utiliza un esquema de versionado en la URL, y que la versión actual documentada es la v0.

Sin embargo, no hay detalles sobre cómo se manejarán futuras versiones (ej. si habrá nuevas rutas como /v1/, cómo se gestionará la obsolescencia de versiones antiguas, o si se utiliza versionado por cabeceras). La ausencia de un *changelog* accesible dificulta que los desarrolladores puedan rastrear cambios, nuevas funcionalidades, correcciones de errores o cambios que rompan la compatibilidad en la API a lo largo del tiempo. La API de docs.crunchybridge.com [6] sí menciona un *changelog* y una política de retrocompatibilidad, pero nuevamente, esto no es directamente

transferible a apidocs.bridge.xyz.

14. Casos de Uso Comunes

La API de Bridge.xyz está diseñada para soportar una variedad de casos de uso que giran en torno a la interconexión de sistemas financieros tradicionales con el mundo de las *stablecoins* y los activos digitales. Algunos de los ejemplos más ilustrativos de su aplicación incluyen:

- **Rampas de Entrada y Salida Fiat-Cripto:** Utilizar el *endpoint* de Transferencias para facilitar a los usuarios la conversión de monedas fiduciarias como USD, Euro o MXN a *stablecoins* (*on-ramp*), y viceversa, la conversión de *stablecoins* de vuelta a fiat depositado en cuentas bancarias (*off-ramp*).[3]
- **Intercambios entre Stablecoins (Stablecoin Swaps):** Emplear la API de Transferencias para realizar intercambios directos entre diferentes tipos de *stablecoins*, por ejemplo, de USDT a USDC, o para mover la misma *stablecoin* entre diferentes blockchains.[3]
- **Cuentas de Depósito Dedicadas para On-Ramp:** Crear Cuentas Virtuales para ofrecer a los clientes finales números de cuenta y ruta bancarios únicos (o IBANs para Euros, CLABEs para MXN) para que puedan recibir depósitos en USD (u otras monedas fiat soportadas) que luego se convierten automáticamente a *stablecoins* y se envían a una *wallet* predefinida.[3]
- **Liquidación Automatizada de Ingresos en Criptomonedas:** Utilizar Direcciones de Liquidación para establecer direcciones de depósito de blockchain permanentes. Cuando estas direcciones reciben criptomonedas (ej. un comerciante recibe USDC en la red Solana), los fondos se convierten y liquidan automáticamente en una moneda fiat (como USD o Euros) a una cuenta bancaria, o a otra *stablecoin* en una *wallet* diferente.[3]
- **Servicios de Carteras Custodia (Custodial Wallets):** Crear y gestionar carteras (*wallets*) en nombre de los usuarios para recibir, almacenar y potencialmente enviar *stablecoins* y otros activos digitales soportados.[3] Bridge se encargaría de la complejidad técnica subyacente.[2]
- **Emisión de Stablecoins Propias:** Aprovechar las capacidades de emisión de Bridge para lanzar *stablecoins* personalizadas o utilizar la *stablecoin* de Bridge (USDB), con la posibilidad de ganar recompensas sobre los activos de reserva.[2]
- **Movimiento Generalizado y Flexible de Fondos:** La API de Transferencias, con su diseño generalizado, permite una gran flexibilidad para orquestar flujos de dinero que pueden ir de fiat a cripto, de cripto a fiat, o entre diferentes activos y redes cripto, adaptándose a necesidades específicas.[14]
- **Pagos Transfronterizos:** Utilizar la infraestructura de Bridge para enviar pagos

internacionales que pueden liquidarse instantáneamente y de manera más económica en comparación con los sistemas tradicionales.[2]

Estos casos de uso demuestran la versatilidad de la API de Bridge.xyz, posicionándola como una herramienta potente para empresas FinTech, plataformas de intercambio, proveedores de servicios de pago y cualquier negocio que busque integrar funcionalidades de *stablecoin* y fiat en sus operaciones. El enfoque principal es tender puentes entre los sistemas financieros tradicionales y el emergente ecosistema de activos digitales, ofreciendo soluciones tanto para la conversión de fondos como para operaciones nativas dentro del espacio cripto.

15. Recursos Adicionales y Soporte

Para facilitar la integración y resolver posibles problemas, Bridge.xyz ofrece varios canales de soporte y recursos documentales a los desarrolladores.

15.1. Soporte al Desarrollador

Según la información disponible [48], Bridge.xyz adopta un enfoque colaborativo para el soporte técnico:

- **Canales Compartidos de Comunicación:** Se establece la práctica de configurar canales compartidos en plataformas de mensajería como Slack o Discord directamente con el equipo del desarrollador. Esto permite una comunicación más fluida y rápida para trabajar conjuntamente en la resolución de problemas de clientes o dudas técnicas.
- **Ejemplo de Interacción:** Si Bridge detecta un problema, como un pago devuelto debido a información bancaria incorrecta, el equipo de Bridge se pondría en contacto a través de estos canales para discutir con el equipo del desarrollador los siguientes pasos, que podrían incluir reintentar el pago, solicitar la actualización de la información de la cuenta, o devolver los fondos a la cuenta de origen.
- **Contacto por Correo Electrón మనసులో మాట: El equipo del desarrollador o sus clientes también pueden contactar directamente al soporte de Bridge a través de la dirección de correo electrónico: support@bridge.xyz.

Este modelo de soporte, especialmente a través de canales de mensajería directa, sugiere un enfoque ágil y potencialmente más eficiente para la resolución de problemas en comparación con los sistemas tradicionales basados únicamente en tickets de soporte.

15.2. Documentación de la API

La documentación principal de la API se encuentra alojada en el sitio apidocs.bridge.xyz. Esta documentación está estructurada en varias secciones principales para guiar a los desarrolladores [48]:

- **Getting Started:** Introducción general a la API.
- **Core API Concepts:** Explicación de los conceptos fundamentales de la API (ver Sección 1.1 de este informe).
- **Products:** Descripción de los diferentes productos ofrecidos a través de la API (ej. Transferencias, Cuentas Virtuales, Direcciones de Liquidación).
- **What we support:** Información sobre *stablecoins*, blockchains y métodos de pago fiduciarios soportados.
- **Compliance Requirements:** Detalles sobre los requisitos de cumplimiento para individuos y empresas.
- **Fees:** Información sobre las comisiones de Bridge y las Comisiones para Desarrolladores.
- **Logistics:** Detalles sobre liquidación, horarios, y días festivos.
- **API Reference:** Una sección que lista los *endpoints* de la API con detalles sobre sus parámetros y respuestas (ejemplos en [28]).
- **Sandbox:** Información sobre el entorno de pruebas para la API.[41]

A pesar de esta estructura, es importante reiterar una observación recurrente a lo largo de este informe: varios fragmentos de investigación indicaron que ciertas páginas o secciones específicas de la documentación de apidocs.bridge.xyz eran "inaccesibles" o que "la información solicitada no estaba disponible en el documento" en el momento de la recopilación de datos (ej. [25] entre otros). Esto podría ser indicativo de una documentación incompleta en ciertas áreas, problemas temporales con el sitio web, o limitaciones de la herramienta de investigación utilizada. Esta situación subraya la importancia de que los desarrolladores consulten siempre la versión más reciente y completa de la documentación directamente en el portal de Bridge.xyz y utilicen los canales de soporte para aclarar cualquier duda o información faltante.

16. Conclusión y Próximos Pasos para Desarrolladores

La API de Bridge.xyz emerge como una plataforma robusta y multifacética diseñada para simplificar la compleja interacción entre las finanzas tradicionales y el creciente ecosistema de *stablecoins* y activos digitales. Sus capacidades principales, que incluyen la orquestación de pagos fiat y cripto, la emisión de *stablecoins* (incluyendo las personalizadas), y la provisión de herramientas para la gestión de clientes

(KYC/KYB), la convierten en una opción atractiva para desarrolladores y empresas que buscan construir o mejorar aplicaciones FinTech. Productos como las Transferencias generalizadas, las Direcciones de Liquidación, las Cuentas Virtuales y los Memos Estáticos ofrecen una considerable flexibilidad para una amplia gama de casos de uso, desde simples rampas de entrada/salida hasta sistemas de pago y liquidación más sofisticados.

Para una integración exitosa y segura, los desarrolladores deben prestar especial atención a varios aspectos clave:

- **Gestión Segura de Claves API:** Dada la autenticación mediante una clave API estática, su protección es primordial.
- **Implementación de Idempotencia:** El uso de la cabecera Idempotency-Key es crucial para prevenir operaciones duplicadas en un entorno financiero.
- **Manejo de Flujos Asíncronos:** Muchas operaciones, como los depósitos de usuarios (tanto fiat como cripto) y las notificaciones vía webhooks, son de naturaleza asíncrona. Las aplicaciones deben estar diseñadas para manejar estos flujos, actualizando su estado y la interfaz de usuario de manera apropiada.
- **Verificación de Firmas de Webhooks:** Es un paso de seguridad no negociable para asegurar la autenticidad de las notificaciones recibidas.

No obstante, este análisis, basado en los materiales de investigación proporcionados, también ha revelado áreas donde la documentación parece ser deficiente o estaba inaccesible. Estas incluyen, de manera crítica:

- **Manejo General de Errores:** Faltan detalles sobre la estructura de los mensajes de error y los códigos de error específicos de la aplicación.
- **Estructuras de Datos Formales:** No se dispone de esquemas o modelos de datos centralizados (como una especificación OpenAPI completa).
- **Límites de Tasa de la API:** Información esencial para el diseño de aplicaciones de producción.
- **Política de Versionado Detallada y Changelog:** Necesarios para gestionar la evolución de la API.
- **Detalles Completos sobre Tipos de Eventos de Webhook y sus Estructuras:** Fundamental para una implementación completa de webhooks.

Dada estas lagunas, se recomienda encarecidamente a los desarrolladores que, además de utilizar la documentación disponible en apidocs.bridge.xyz, se pongan en contacto directo con el equipo de soporte de Bridge.xyz para obtener la información más actualizada y completa sobre estos aspectos.

Como **próximos pasos** para un desarrollador interesado en integrar la API de Bridge.xyz, se sugiere la siguiente secuencia:

1. **Crear una Cuenta de Desarrollador:** Registrarse en dashboard.bridge.xyz.
2. **Obtener Claves API:** Generar claves API tanto para el entorno de Sandbox (pruebas) como para Producción, almacenándolas de forma segura.
3. **Familiarizarse con el Entorno Sandbox:** Realizar pruebas exhaustivas de los flujos principales, comenzando por:
    - La API de Clientes (creación, actualización, manejo de estados KYC).
    - Los diferentes tipos de Transferencias (fiat-a-stablecoin, stablecoin-a-fiat, stablecoin-a-stablecoin), prestando especial atención al manejo de las source_deposit_instructions y los blockchain_memos.
4. **Implementar la Lógica de Autenticación e Idempotencia** desde el inicio del desarrollo.
5. **Configurar y Probar Webhooks** (en la medida que la información sobre eventos lo permita), incluyendo la verificación de firmas.
6. **Consultar al Soporte de Bridge.xyz** para aclarar dudas y obtener información sobre las áreas documentales deficientes identificadas.

Siguiendo estos pasos y manteniendo una comunicación abierta con el proveedor de la API, los desarrolladores estarán mejor posicionados para aprovechar el potencial de la API de Bridge.xyz en sus proyectos.

Fuentes citadas

1. Getting Started - Bridge, acceso: mayo 30, 2025, https://apidocs.bridge.xyz/docs/getting-started
2. Bridge | Stablecoin Infrastructure and APIs for Developers, acceso: mayo 30, 2025, https://www.bridge.xyz/
3. API Summary - Bridge, acceso: mayo 30, 2025, https://apidocs.bridge.xyz/docs/api-summary
4. Core API Concepts - Bridge.xyz, acceso: mayo 30, 2025, https://apidocs.bridge.xyz/docs/core-api-concepts
5. Quick Start - Bridge, acceso: mayo 30, 2025, https://apidocs.bridge.xyz/docs/quick-start
6. Getting started - Crunchy Bridge, acceso: mayo 30, 2025, https://docs.crunchybridge.com/api-concepts/getting-started
7. Customers API - Bridge, acceso: mayo 30, 2025, https://apidocs.bridge.xyz/docs/customers-api
8. apidocs.bridge.xyz, acceso: mayo 30, 2025, https://apidocs.bridge.xyz/docs/endorsements
9. KYC Links for New Customers - Bridge, acceso: mayo 30, 2025, https://apidocs.bridge.xyz/docs/kyc-links

10. Additional Requirements for Existing Customers - Bridge, acceso: mayo 30, 2025, https://apidocs.bridge.xyz/docs/additional-requirements-for-existing-customers
11. Rejection Reasons - Bridge, acceso: mayo 30, 2025, https://apidocs.bridge.xyz/docs/rejection-reasons
12. External Accounts - Bridge, acceso: mayo 30, 2025, https://apidocs.bridge.xyz/docs/external-accounts
13. acceso: diciembre 31, 1969, https://apidocs.bridge.xyz/docs/external-accounts-api
14. Transfers - Bridge, acceso: mayo 30, 2025, https://apidocs.bridge.xyz/docs/transfers-1
15. Fiat to Stablecoin (Crypto On-Ramp) - Bridge, acceso: mayo 30, 2025, https://apidocs.bridge.xyz/docs/fiat-to-stablecoin-crypto-on-ramp
16. Stablecoin to Fiat (Crypto Off-Ramp) - Bridge, acceso: mayo 30, 2025, https://apidocs.bridge.xyz/docs/stablecoin-to-fiat-crypto-off-ramp
17. Stablecoin to Stablecoin - Bridge, acceso: mayo 30, 2025, https://apidocs.bridge.xyz/docs/stablecoin-to-stablecoin
18. Transfer Features - Bridge, acceso: mayo 30, 2025, https://apidocs.bridge.xyz/docs/transfer-features
19. apidocs.bridge.xyz, acceso: mayo 30, 2025, https://apidocs.bridge.xyz/docs/fetching-transfers
20. apidocs.bridge.xyz, acceso: mayo 30, 2025, https://apidocs.bridge.xyz/docs/deleting-transfers
21. Liquidation Address - Bridge, acceso: mayo 30, 2025, https://apidocs.bridge.xyz/docs/liquidation-address
22. acceso: diciembre 31, 1969, https://apidocs.bridge.xyz/docs/liquidation-drains
23. acceso: diciembre 31, 1969, https://apidocs.bridge.xyz/docs/liquidation-address-api
24. Virtual Accounts - Bridge, acceso: mayo 30, 2025, https://apidocs.bridge.xyz/docs/virtual-accounts
25. acceso: diciembre 31, 1969, https://apidocs.bridge.xyz/docs/virtual-accounts-api
26. Static Memos - Bridge, acceso: mayo 30, 2025, https://apidocs.bridge.xyz/docs/static-memos
27. Webhook Event Signature Verification - Bridge.xyz, acceso: mayo 30, 2025, https://apidocs.bridge.xyz/docs/webhook-event-signature-verification
28. Create a webhook endpoint - Bridge, acceso: mayo 30, 2025, https://apidocs.bridge.xyz/reference/post_webhooks
29. acceso: diciembre 31, 1969, https://apidocs.bridge.xyz/docs/webhooks-overview
30. acceso: diciembre 31, 1969, https://apidocs.bridge.xyz/docs/webhooks-event-types
31. apidocs.bridge.xyz, acceso: mayo 30, 2025, https://apidocs.bridge.xyz/docs/event-structure
32. Developer Fees - Bridge, acceso: mayo 30, 2025, https://apidocs.bridge.xyz/docs/developer-fees
33. acceso: diciembre 31, 1969, https://apidocs.bridge.xyz/docs/error-handling
34. apidocs.bridge.xyz, acceso: mayo 30, 2025, https://apidocs.bridge.xyz/docs/handling-deposit-errors

35. acceso: diciembre 31, 1969, https://apidocs.bridge.xyz/docs/errors
36. acceso: diciembre 31, 1969, https://apidocs.bridge.xyz/docs/error-handling-overview
37. API basics - Bridge API Documentation, acceso: mayo 30, 2025, https://docs.bridgeapi.io/docs/basics
38. apidocs.bridge.xyz, acceso: mayo 30, 2025, https://apidocs.bridge.xyz/reference
39. acceso: diciembre 31, 1969, https://apidocs.bridge.xyz/docs/rate-limits
40. acceso: diciembre 31, 1969, https://apidocs.bridge.xyz/docs/rate-limiting-policy
41. Sandbox - Bridge, acceso: mayo 30, 2025, https://apidocs.bridge.xyz/docs/sandbox
42. Rate limits - Bridge API Documentation, acceso: mayo 30, 2025, https://docs.bridgeapi.io/v2019.2.18/docs/rate-limits
43. Bridge Terms of Use | API Policy, acceso: mayo 30, 2025, https://www.getbridge.com/bridge-terms-of-use/
44. acceso: diciembre 31, 1969, https://apidocs.bridge.xyz/changelog
45. acceso: diciembre 31, 1969, https://apidocs.bridge.xyz/docs/versioning
46. acceso: diciembre 31, 1969, https://apidocs.bridge.xyz/docs/api-versioning
47. Update a single customer object - Bridge, acceso: mayo 30, 2025, https://apidocs.bridge.xyz/reference/put_customers-customerid
48. Support - Bridge, acceso: mayo 30, 2025, https://apidocs.bridge.xyz/docs/support
49. acceso: diciembre 31, 1969, https://apidocs.bridge.xyz/docs/transfers-api