Boletín CIMPRA N°544

21 de agosto de 2025

Comunicación A 8206
Viaje con QR (VQR) /
Documentación Técnica

BANCO CENTRAL
DE LA REPÚBLICA ARGENTINA

Boletín CIMPRA N° 544

Comunicación A 8206
Viaje con QR (VQR) /
Documentación Técnica

Introducción

La Comunicación A 8206 del 27/2/2025 estableció que el Viaje con QR (VQR) podrá estar disponible para uso por parte del público a partir del 12/05/25, pudiendo las billeteras digitales interoperables ofrecerlo a partir de esa fecha.

En ese marco, en el ámbito de la CIMPRA se han llevado a cabo reuniones con todos los participantes del ecosistema tales como las asociaciones de bancos, la Cámara Argentina Fintech, administradores QR de transporte, billeteras, etc. con el objetivo de cumplir con los plazos normativos y los desarrollos tecnológicos necesarios.

Se incluye a continuación la documentación técnica desarrollada y consensuada, la cual configura un estándar único y una solución abierta e interoperable para los operadores del sistema de transporte, administradores QR, billeteras digitales interoperables y usuarios.

La documentación contempla, entre otras cuestiones, la configuración inicial conjunta que deben realizar billeteras, operadores del sistema de transporte y los administradores QR de transporte.

Toda mejora funcional, operativa, gestión de fraude, etc. será incorporada y publicada en sucesivos Boletines CIMPRA.

Solución QR Transporte

Índice

1. Características generales

1.1. Objetivo

El presente documento tiene como objetivo especificar la solución técnica para que los usuarios del sistema de transporte puedan pagar sus viajes utilizando un código QR generado en su aplicación de billetera digital interoperable registrada en el Banco Central de la República Argentina (en adelante, también "billetera(s)" o "BDI").

En transporte el QR fue seleccionado como un medio para iniciar pagos porque es una alternativa inclusiva para todas las personas que no usan tarjetas o que prefieren ir siguiendo día a día su gasto en transporte. También aplica para usuarios que quieran pagar con el teléfono, pero no tengan tecnología con NFC (se estima que el 70% de los teléfonos en Argentina no tiene esta funcionalidad).

El QR para el pago de transporte será una solución abierta para que las billeteras que deseen puedan incorporarlo en beneficio de sus usuarios.

Adicionalmente, el objetivo es construir un sistema estándar único y simple para todos los actores del sistema y de fácil implementación para los medios de transporte que deseen incorporarlo.

1.2. Alcance

El alcance del proyecto es nacional, independientemente de la jurisdicción. Se busca establecer un estándar único y ágil que pueda ser adoptado tanto para todas las billeteras como para la totalidad de los medios de transporte (colectivos, trenes, subtes, etc.).

1.3. Consideraciones del sistema de transporte

- Es un sistema abierto que permite integrar operadores de transporte y billeteras.
- Permite a los actores del sistema validar los códigos QR.
- El código QR se puede generar desde los dispositivos billetera aun cuando no tengan conexión a internet de forma permanente.
- Los validadores, una vez ejecutados los controles de seguridad, tienen que habilitar el acceso de forma rápida para permitir el flujo de los usuarios en el sistema de transporte.
- Permite que las transacciones se ejecuten de forma asincrónica, una vez que los validadores tengan conexión a internet.
- Se realiza una transferencia diaria en donde las billeteras liquidan los viajes aprobados al administrador QR de transporte (en adelante, también "administrador QR").

2. Actores del ecosistema

2.1. Operador del sistema de transporte (operador)

- Está a cargo de la colocación y mantenimiento de los validadores.
- Tiene el rol de asegurarse la incorporación del *software* dentro del *hardware*.
- Proporciona las tarifas a cobrar del sistema.

2.2. Administrador QR de transporte (administrador QR)

- Es el encargado de gestionar los pagos, generando los pedidos de autorización de viajes con los QR a las billeteras.
- Ejecuta las transferencias al final del día.
- Gestiona las listas de negación por cada billetera del sistema.
- Está a cargo de la recuperación de la deuda de cada billetera.

2.3. Billetera

- Aplicación móvil registrada ante el BCRA como billetera digital interoperable donde el usuario genera el QR para ser presentado ante el validador.
- Aprueba o rechaza la solicitud de viaje.
- Gestiona su lista de denegación.
- Concilia y liquida los viajes adeudados.

2.4. Usuario

- Son los propietarios del dispositivo móvil con la aplicación billetera instalada.
- Poseen fondos en sus cuentas para poder comprar el viaje.

3. Resumen funcional

3.1. Diagrama

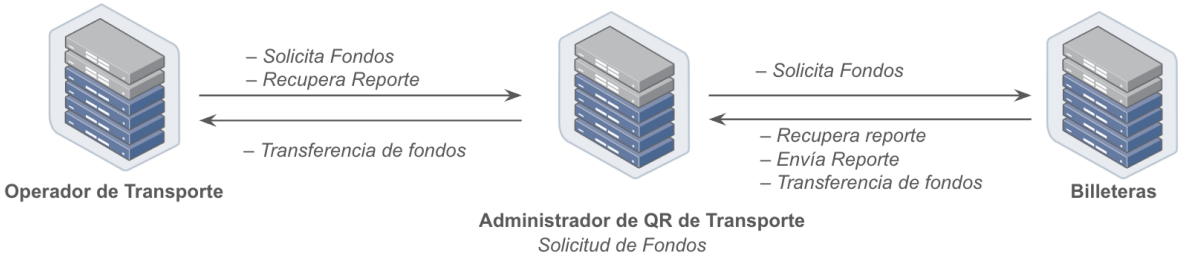
3.2. Aceptación

- Los usuarios de las billeteras generan un código “QR comprador” (*consumer presented QR*) para exponer frente a los validadores del sistema de transporte con el fin de efectuar un viaje.
- Los validadores verifican, mediante un esquema de seguridad (robusto y estándar) presentado en este documento, la integridad del código QR presentado por el usuario y la presencia de la cuenta del usuario (*Account ID*) en las listas de denegación, para aceptar o rechazar un viaje.
- Los validadores acumulan los viajes válidos e inválidos para ser transferidos luego al servicio de procesamiento del administrador QR de transporte.

3.3. Procesamiento

- El administrador QR envía a las billeteras cada uno de los viajes confirmados por el validador para ser autorizados.
- Los servicios de procesamiento de las billeteras responden a los pedidos de autorización del administrador QR.
- El administrador QR debe agregar en la lista de denegación para evitar posteriores viajes- al usuario de una billetera cuando reciba como respuesta alguno de los siguientes *status_code*:
    - *APPROVED_OVERLIMIT*
    - *APPROVED_HIGH_RISK*
    - *REJECTED_DENY_LIST*
- El administrador QR es el encargado de propagar la lista de denegación hacia los validadores del operador de sistema de transporte.

3.4. Conciliación



El administrador QR envía a las billeteras y al operador de transporte un reporte conciliatorio diario con el detalle de los viajes realizados por los usuarios.

3.5. Liquidación

Diariamente el administrador QR presenta un reporte y solicita a las billeteras fondos a transferir.

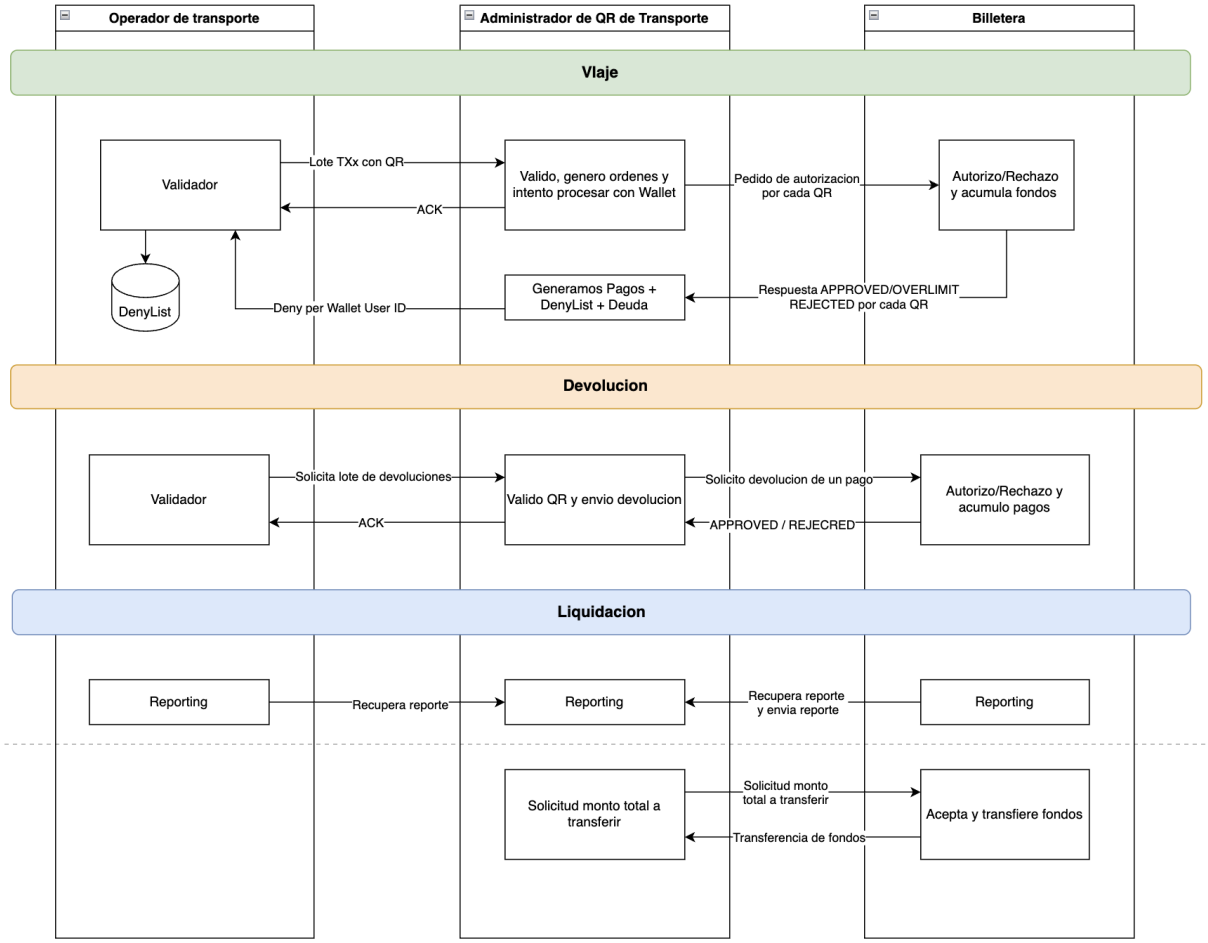
Posteriormente a la liquidación las billeteras deben compartir un reporte indicando el desglose de dicha liquidación transaccional para que el administrador QR pueda hacer controles conciliatorios.

4. Esquema general de la solución

4.1. Diagrama

La solución tiene 4 fases o etapas:

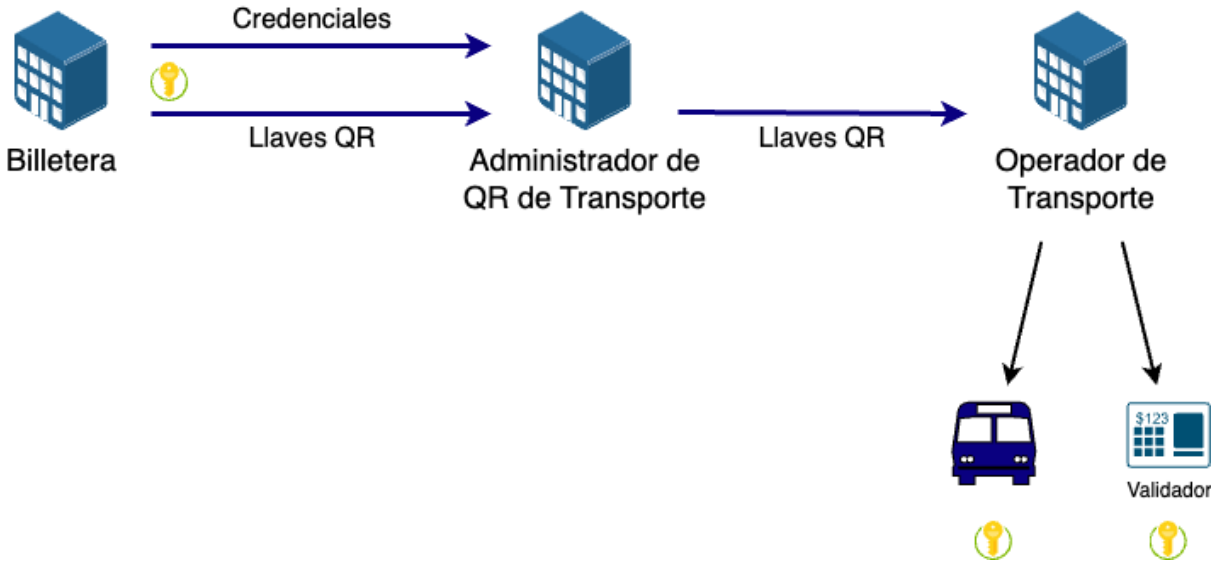
- Configuración: ocurre antes de iniciar la operación del sistema.
- Lectura de códigos QR: ocurre cuando los usuarios intentan viajar presentan sus códigos QR a los validadores.
- Procesamiento: ocurre cuando los validadores tienen conectividad y envían al operador de transporte los viajes para iniciar el circuito de autorización, tal lo descripto en la primera parte del siguiente diagrama.
- Liquidación: ocurre una vez terminado el día y se concreta con un envío de fondos.



4.2. Configuración inicial

La billetera, el operador de transporte y el administrador QR deberán realizar una configuración inicial conjunta para garantizar la segura interacción de sus sistemas:

- Inyección de llaves criptográficas para operar de forma segura las tramas de los códigos QR.
- Provisión de credenciales para operar los sistemas de procesamiento y liquidación.

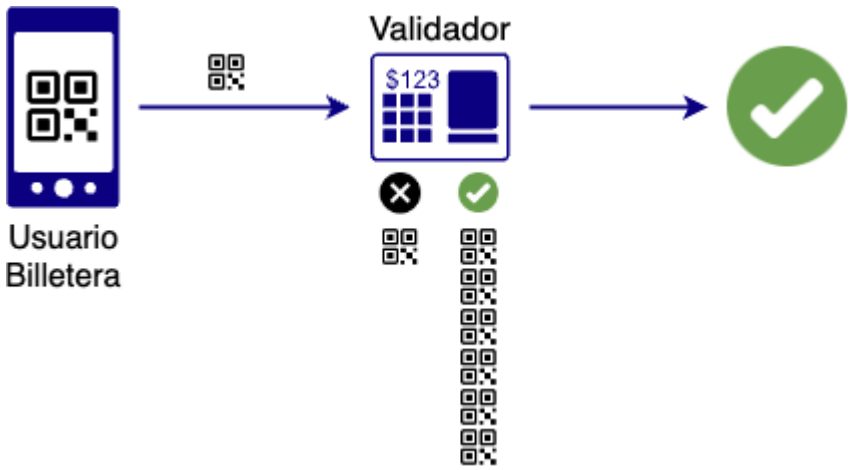


Más detalle en el Anexo Técnico: Seguridad.

4.3. Lectura

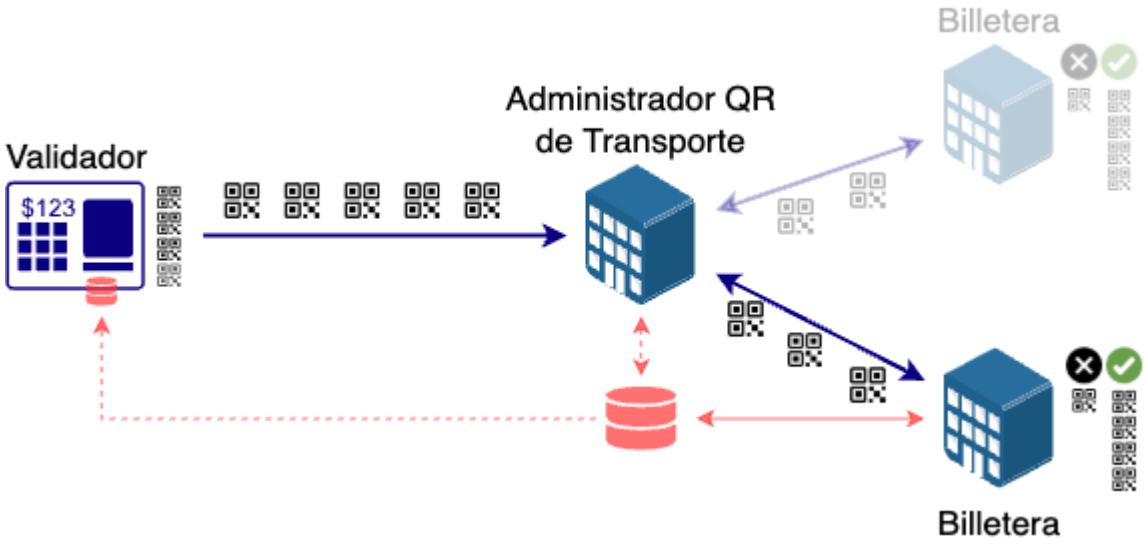
4.3.1. Generación y validación de QR

- El usuario abre su aplicación de billetera y solicita la generación de un código QR:
    - La generación de este código debe ser posible a pesar de que el usuario no tenga conexión a internet en el momento que desea realizar un viaje.
    - La aplicación billetera puede elegir no generar un código QR si no desea que su usuario viaje.
- El usuario presenta el código QR ante el validador para autorizar un viaje por una tarifa previamente acordada.
- El validador realiza validaciones y, si son satisfactorias, debe habilitar al usuario a realizar el viaje.
- El validador almacena los datos del código QR hasta poder transmitir esta información al administrador QR.



4.4. Procesamiento

4.4.1. Viajes



- El administrador QR recibe de los validadores los resultados de las lecturas de los códigos QR utilizados con un código único de viaje:
    - Los códigos QR deberán ser validados para garantizar su integridad.
    - Los códigos QR que fallen las validaciones serán apartados para realizar notificaciones a la billetera con el fin de encontrar posibles errores de aceptación o procesamiento.
- Los sistemas del administrador QR envían al sistema de procesamiento de la billetera cada uno de los viajes para ser autorizados. El pedido de autorización debe incluir:
    - El código QR que fue capturado por el validador.
    - La fecha y hora en la que se escaneó el QR.

- La fecha y hora en la que se creó el viaje en el administrador.
- Un ID único para identificar el viaje en el administrador (id).
- Un ID único para identificar el viaje en el operador (*external_reference*).
- El monto del viaje.
- Una descripción del concepto del viaje.
- Operador de transporte del viaje.
- Información del dispositivo validador.

- El sistema de procesamiento de la billetera recibe la solicitud de autorización del administrador QR y ejecuta el proceso de autorización para generar un pago:
  - Realiza validaciones de integridad del QR y verifica si el usuario está en la lista de denegación.
  - Ejecuta la autorización respondiendo de forma sincrónica con los estados:
    - aprobado (*APPROVED)*
    - rechazado (*REJECTED*)
  - El resultado de la autorización también es notificado al administrador mediante un servicio de notificación.

4.4.2. Devoluciones

En caso de querer realizar la devolución de un viaje a un usuario, el operador del sistema de transporte tiene la posibilidad de ejecutar una operación de devolución.

El administrador QR puede enviar al sistema de procesamiento de la billetera una solicitud de devolución por el monto total del viaje.

Si el *Account ID* que debe recibir la devolución no está en la lista de denegación, el usuario mostrará un QR válido para transporte y el validador la procesará siempre que esté configurado en una modalidad que permita generar una devolución.

Por el contrario, si el *Account ID* está en la lista de denegación y la billetera no permite, por lo tanto, la generación del código QR válido para transporte, al efecto de que el usuario solicite la devolución le puede generar un código QR con la funcionalidad *Deny for transit*. Este código QR va a ser aceptado por el validador, siempre que esté configurado en una modalidad que permita generar devoluciones.

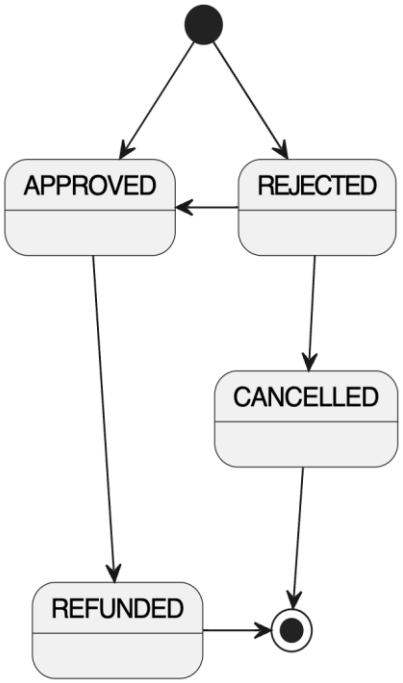
4.4.3. Estados de un viaje

Los viajes tienen dos atributos para definir estados:

- *status*
- *status_code*

El campo *status* es el atributo principal; todo viaje que retorne un estado *APPROVED* debe ser honrado por la billetera al administrador QR.

- *APPROVED*: el viaje fue aprobado.
- *REJECTED*: el viaje fue rechazado.
- *REFUNDED*: el viaje fue reembolsado.
- *CANCELLED*: un viaje anteriormente rechazado está cancelado.



El campo *status_code* se utiliza para tener un detalle del *status* y sirve para modelar particularidades de las aprobaciones o rechazos.

- *APPROVED*: el viaje fue aprobado.
- *APPROVED_OVERLIMIT*: el viaje fue aprobado, pero excede el límite del usuario. La *Account ID* debe ser agregada a la lista de denegación.

- *APPROVED_HIGH_RISK*: el viaje fue aprobado pero la billetera no desea aceptar nuevos viajes de este usuario por potenciales riesgos. La *Account ID* debe ser agregada a la lista de denegación.
- *REJECTED_DENY_LIST*: el viaje fue rechazado porque la *Account ID* se encuentra presente en la lista de denegación de la billetera. La *Account ID* debe ser agregada a la lista de denegación del validador.
- *REJECTED_QR_INTEGRITY*: el viaje fue rechazado por fallas de integridad, seguridad o inconsistencias del código QR.
- *REJECTED_QR_INVALID_FORMAT*: el viaje fue rechazado porque el QR no tiene un formato válido.
- *REJECTED_QR_EXPIRED*: el viaje fue rechazado porque se utilizó un QR expirado.
- *REJECTED_QR_DUPLICATED*: el viaje fue rechazado porque el código QR ya fue utilizado anteriormente en el mismo validador.
- *REJECTED_EXCEEDED_MAX_AMOUNT*: el viaje fue rechazado porque excede el monto máximo permitido.
- *REJECTED_AFTER_DEADLINE*: el viaje fue enviado a autorizar posteriormente a la fecha límite definida en los parámetros de riesgo.
- REFUNDED: el viaje fue devuelto.
- CANCELED: un viaje previamente rechazado fue devuelto.

4.4.4. Lista de denegación

La lista de denegación será el método que tienen las billeteras y el administrador QR para denegar el viaje a un *Account ID* específico. La lista de denegación será poblada a través de estímulos de la billetera que pueden ser de dos tipos:

- Según respuestas sobre viajes (*ride.result.status_code*):
    - *APPROVED_OVERLIMIT*
    - *APPROVED_HIGH_RISK*
    - *REJECTED_DENY_LIST*
- Utilizando los servicios de denegación que proporciona el administrador.

Una vez que se da de alta a un *Account ID* en la lista de denegación del administrador QR, este debe enviar de forma inmediata la información al operador de transporte para ser incluida en el validador. El tiempo máximo que tienen el administrador QR y operador de transporte para

actualizar un parámetro de la lista de denegación está indicado en el parámetro de riesgo *Refresh rate deny list*.

El validador no debe permitir viajar a un *Account ID* que esté en la lista de denegación.

La billetera no debe permitir generar QR válidos para transporte a un *Account ID* que esté en la lista de denegación, salvo en el uso de la funcionalidad *Bypass deny list*.

La lista de denegación que tiene el validador implementará un tiempo de expiración (*TTL*) de registros para evitar llegar a los límites de almacenamiento.

El tiempo configurado para la expiración debe ser ajustado según lo establecido en el parámetro de riesgo *TTL de la deny list*.

Si la billetera permite la generación de un código QR válido para transporte posterior al *TTL* de la lista de denegación del validador, la billetera debe honrar el pago de forma forzada, independientemente de que el viaje tenga una respuesta de aprobado o rechazado por parte de la billetera.

En caso de que el administrador QR y el operador de transporte no puedan actualizar la lista de denegación en un validador dentro del plazo de 2 horas establecido en el parámetro de riesgo "*TTL* de la lista de denegación", y se libere un viaje con un código QR, cuya *Account ID* debería haber estado en la lista de denegación, no debe ser honrado por la billetera cuando el resultado de la autorización haya sido *APPROVED_OVERLIMIT* o *REJECTED_DENY_LIST*.

Una vez que el *Account ID* saldó su deuda con la billetera, la billetera puede intentar quitarlo de la lista de denegación utilizando los servicios de denegación.

Los servicios de denegación pueden rechazar la petición de quitar al usuario de la lista de denegación si el *Account ID* posee deuda con el administrador de QR. En caso de rechazo, la billetera no debe quitar al usuario de su lista de denegación local. Ver Gestión de Deuda/ Servicio de notificación de recupero de deuda.

4.4.5. Funcionalidad *Deny for transit*

Esta funcionalidad permite a la billetera generar un código QR que no es válido para su utilización en el sistema de transporte definido en este documento. Los validadores deben denegar su utilización, salvo que se trate de una devolución. Esta funcionalidad está definida en la Tabla de QR *Feature Flags*.

4.4.6. Funcionalidad *Bypass deny list*

La funcionalidad permite a la billetera dejar viajar a una cuenta a pesar de estar en las listas de denegación, teniendo que honrar este viaje respondiendo siempre con una aprobación.

Para activar la funcionalidad, la billetera genera un código QR con la funcionalidad *Bypass deny list* (ver Tabla QR *Feature flags*). El validador detecta la funcionalidad en el código QR y saltea la validación en la lista de denegación.

El administrador QR recibe el QR con la funcionalidad *Bypass deny list* activada y envía el pedido de autorización a la billetera indicando que se utilizó este tipo de operatoria. La billetera debe aprobar el viaje de forma obligatoria por haber enviado el código QR con la funcionalidad *Bypass deny list* activada.

Si la cuenta tiene deuda con el administrador QR, se utilizará el servicio de reintento de viajes rechazados para cobrar los viajes adeudados, los cuales deben ser obligatoriamente aprobados por la billetera.

Además de esta información estar presente en la mensajería del servicio de procesamiento, estará presente en reporte conciliatorio.

En el caso que se dé una devolución del viaje que fue enviado por la billetera al administrador QR con esta funcionalidad, únicamente se procederá a devolver el viaje en cuestión. El proceso de cobro de deuda continuará su flujo tal cual se mencionó anteriormente.

4.5. Liquidación

4.5.1. Reportes conciliatorios

El administrador QR genera para la billetera un reporte diario con el detalle de los viajes realizados por los usuarios.

El reporte está ordenado por el atributo *processed at*, que indica el momento en el cual la billetera respondió a la autorización de un viaje. Contiene todos los viajes aceptados, rechazados, devueltos y cancelados dentro de un período de 24 horas (desde las 00 del YYYMMDD hasta las 23:59:59 de YYYYMMDD) del atributo *processed_at*. El archivo está delimitado por punto y coma ";".

Los reportes tienen un ID único por día con la siguiente forma {*YYYYMMDD*}-{*WALLET_ID*}_{*CURRENCY*}_*report*.csv:

- *YYMMDD*: fecha del reporte, ejemplo 26/12/2004.
- *WALLET_ID*: identificador de billetera, ejemplo 33535.
- *CURRENCY*: moneda del reporte, ejemplo ARS.

El reporte incluye los siguientes campos/columnas:

- *ride_id*: ride ID generado por el administrador QR, ejemplo ride_01JQ97WQ8GMK9YS0B3V0D02SKX.
- *payment_id*: ID generado por la billetera en la autorización del viaje, ejemplo payment_100277356.
- *external_reference*: identificador generado por el validador, ejemplo 1000090080041123.
- *net_amount*: monto neto del viaje expresado en decimales, por ejemplo 1500.50 para $1500.50.
- *gross_amount*: monto bruto del viaje expresado en decimales, por ejemplo 1600.13 para $1600.13.
- *fee*: monto de la comisión expresado en decimales, por ejemplo 0.0005 para 0.05%
- *currency*: moneda de pago del viaje, por ejemplo ARS para pesos argentinos.
- *status*: estado del viaje, ejemplo *APPROVED* (como está definido en el punto 4.4.3. Estados de un viaje).
- *status_code* detalle de estado del viaje, ejemplo *APPROVED_OVERLIMIT* (como está definido en Estados de un viaje).
- *issuer_id*: dato del emisor que llegó en el QR generado por la billetera, por ejemplo 33535.
- *transport_operator_id*: ID del operador de transporte, por ejemplo, emova.
- *debt*: *flag* para indicar si es por recupero de deuda, por ejemplo 1 para cuando es recupero, 0 cuando no es recupero de deuda.
- *forced*: *flag* para indicar si la autorización del viaje es forzada, por ejemplo 1 cuando es forzada, 0 cuando no es forzada.
- *feature_flags*: *feature flags* del código QR, por ejemplo 00000010 (para la funcionalidad *Bypass deny list*)
- *scanned_at*: fecha de escaneo del QR en el validador, por ejemplo 2025-06-16T23:01:17.301Z.
- *created_at*: fecha de creación del QR, por ejemplo 2025-06-16T23:01:17.301Z.
- *processed_at*: fecha de envío a autorizar el viaje por el administrador QR, por ejemplo 2025-06-16T23:01:17.301Z.

La billetera envía al administrador QR un reporte diario, posterior a la transferencia de fondos, con el detalle de la liquidación transaccional. Este reporte debe estar ordenado por fecha y hora de procesamiento, y debe contener la siguiente información:

- *ride_id*: identificador del viaje del administrador (*ride_id*).
- *payment_id*: identificador del viaje generado por la billetera (*result.payment_id*)
- *net_amount*: monto neto del viaje expresado en centavos, por ejemplo 150050 para $1500.50.
- *fee*: monto de la comisión expresado en decimales, por ejemplo 0.0005 para 0.05%
- *gross_amount*: monto bruto del viaje expresado en centavos, por ejemplo 160013 para $1600.13.
- *status_code:* detalle de estado del viaje, ejemplo *APPROVED_OVERLIMIT* (como está definido en Estados de un viaje).
- *processed_at*: fecha en la que llegó el pedido de autorización a la billetera, por ejemplo 2025-06-16T23:01:17.301Z. Si el viaje cambia de estado, este campo tiene que reflejar la fecha del último cambio de estado.

4.5.2. Transferencia de fondos

Al finalizar el día el administrador QR:

- Envía a las billeteras y al operador de transporte un reporte conciliatorio diario con el detalle de los viajes realizados por los usuarios. Este reporte permite calcular la sumatoria de los fondos que corresponden ser transferidos por cada billetera.
- Envía un reporte de solicitud de fondos por el total de viajes realizados a cada billetera.
- Recibe una transferencia de fondos por parte de cada billetera.

La solicitud de transferencia de fondos debe ser recibida por el servicio de transferencias de la billetera, con el siguiente contenido:

- Identificador único de la solicitud de transferencia diaria (único por día), por ejemplo 20241226_33535_ARS para el día 26/12/2024 billetera 335.35 en pesos.
- Monto bruto a transferir expresado en decimales, por ejemplo 12345.67 para $12345.67.
- Monto neto a transferir expresado en decimales, por ejemplo 493.83 para $ 493,83.
- *Fee* acordado expresado en decimales, por ejemplo 0.0005 para 0,05%.
- Moneda en la cual está expresado el monto a transferir, ejemplo ARS para pesos argentinos.

Esta solicitud debe ser confirmada por el servicio y será reintentada en caso de no obtener confirmación.

Los fondos transferidos al administrador QR por parte de la billetera incluyen todos los viajes aceptados del día, descontando comisiones y devoluciones.

En caso de existir diferencias, se realizará un proceso de reclamos a posteriori a definir.

5. Gestión de deuda

5.1. Aprobación forzada

La aprobación forzada se da cuando se utiliza la funcionalidad *Bypass deny list* o cuando el administrador QR no puede obtener de la billetera una respuesta sobre un viaje y se hayan excedido los tiempos máximos definidos en los parámetros de riesgo.

5.2. Reintento de deuda con aprobación forzada

El administrador QR intentará cobrarle a la billetera todos aquellos viajes rechazados por no honrar el estándar o sin respuesta, reintentando hasta que sean aprobados por la billetera.

Los reportes conciliatorios contienen atributos necesarios para identificar los viajes de aprobación forzada, tanto de la funcionalidad *Bypass deny list* como de los viajes rechazados o sin respuesta.

5.3. Servicio de reintentos para viajes rechazados

Se utiliza en los casos de reintento de cobro de viajes oportunamente rechazados por la billetera y pendientes de saldar al administrador QR.

El administrador QR puede reintentar el cobro de viajes con *status_code: REJECTED_DENY_LIST*

El reintento por parte del administrador QR se puede dar en los siguientes escenarios:

- La autorización del viaje es rechazada en el primer intento por la billetera y es reintentado posteriormente por el administrador QR durante el plazo y periodicidad establecidos en los parámetros de riesgo *"Cantidad de días y veces de reintentos para recupero de deuda"*.

- La billetera solicita la baja de la *Account ID* deudora de la lista de denegación y el administrador QR reintenta el cobro de deuda registrada.

- El administrador QR recibe un viaje con la funcionalidad *Bypass deny list* de un *Account ID* que presenta deuda. El administrador QR reintenta el cobro de la deuda registrada.

- La periodicidad y duración del proceso de reintentos de viajes rechazados queda definido en parámetros de riesgo. Al finalizar este proceso enviará un evento a la billetera indicando si se pudo o no cobrar la deuda.

Endpoints: /denylist, */events*.

5.4. Servicio recupero de deuda

En el caso de que un *Account ID* deje de presentar mora en la billetera, ésta podrá solicitar la eliminación de la cuenta de la lista de denegación.

Al iniciar el proceso de eliminación de la cuenta de la lista de denegación, el administrador QR revisará si la *Account ID* registra deuda con él.

En caso de no existir deuda, la cuenta será quitada de la lista de denegación, recibiendo la billetera una respuesta satisfactoria por parte de los servicios de denegación.

Si la cuenta tiene deuda, se denegará la petición y se iniciará un proceso de recupero de esta deuda utilizando el servicio de reintentos para viajes rechazados.

Una vez saldada la deuda, el administrador QR enviará una notificación (*account_debt_charged*) a la billetera informando que puede volver a intentar eliminar la cuenta en los servicios de denegación.

Si la deuda del *Account ID* con el administrador QR no hubiera podido ser saldada en el plazo límite establecido para efectuar reintentos, se enviará una notificación (*account_debt_pending*) a la billetera informando que no se pudo recuperar la deuda.

En el caso que el *Account ID* vuelva a tener saldo en la billetera, la billetera debe iniciar nuevamente el proceso de eliminación de la cuenta de la lista de denegación.

Endpoints: */denylist*, */events*.

5.5. Disputas

Las billeteras deben transferir el monto que el administrador de QR de transporte les solicite en el proceso de solicitud de fondos.

Esta solicitud estará previamente avalada por un reporte conciliatorio donde se detallarán todos los viajes con sus estados.

En caso que la billetera quiera disputar lo hará en marco de los contratos entre administrador QR y billeteras, con los SLA preestablecidos.

En caso que se detectara un funcionamiento incorrecto por parte de la billetera, el administrador QR puede iniciar una disputa por las transacciones involucradas.

5.6. Responsabilidad compartida

El BCRA ha definido que los viajes impagos no pueden ser trasladados a los transportistas y deben ser absorbidos por administrador QR y/o billetera según corresponda de acuerdo con la responsabilidad en cada caso.

Adicionalmente, los viajes impagos deben ajustarse a lo dispuesto en el presente Boletín y complementarios, tal como lo referido al recupero de deuda y permanencia en las listas de denegación.

La responsabilidad frente a los viajes realizados -tanto con códigos QR generados *on line* como *off line*- que al momento de ser debitados no tengan saldo en la cuenta del usuario recaerá en billetera o administrador QR según corresponda, de acuerdo con la siguiente regla:

1. El 100% de los viajes aceptados por la billetera deberán ser honrados en el pago por la billetera, excepto por los casos que encuadren en el punto 2.

2. En los *status code* en los cuales la billetera decide aceptar el viaje, pero solicita al administrador QR que incorpore al usuario a la lista de denegación, éste tiene un plazo máximo de 2 horas para actualizar la lista de denegación. Los viajes impagos deberán ser afrontados por el administrador QR o billetera conforme a la siguiente regla:

    a. Durante esas 2 horas rige la responsabilidad compartida: hasta 2 viajes por usuario la billetera se hará cargo del pago (este parámetro es fijado inicialmente y podrá ser modificado con el acuerdo de la industria en función del análisis de la evolución del producto). Asimismo, cada billetera podrá aplicar un tope monetario en función de la evaluación y calificación del usuario en casos de tarifas que por sus montos los ameriten. Superados los 2 viajes por usuario en esas 2 horas, de los viajes impagos se hará cargo el administrador QR.

    b. Superado ese límite temporal de 2 horas: la responsabilidad recaerá sobre el administrador QR.

6. Anexo Técnico

6.1. Código QR

En la billetera, los usuarios podrán generar un QR basado en el estándar *EMVCo Consumer Presented QR* el cual podrá ser escaneado en el molinete/validador para realizar el pago del transporte.

El usuario no requiere tener conexión a internet para la generación del QR, que tendrá la validez de definida en el punto 6.3. Parámetros de riesgo.

La estructura del QR se define en base a la siguiente especificación:

Tag **(Hexa)**		**Descripción**	***Length*** **(bytes)**	**Tipo**	**Mandatorio/ Opcional**	**Observación**
85		*QR Code Format Version*	5	an	M	Siempre CPV01
61		*Application Template*	xxx	var	M	-
	4F	*Wallet ID*	5	b (an)	M	Identificador de la billetera en el ecosistema de QR de transporte, el valor es un código de BCRA de emisores. Encodeado en formato binario ASCI-HEX el valor es un código de BCRA de emisores. Encodeado en ASCII ISO-8859-1. Para esta versión es un binario que se interpreta como an. Ejemplo: para el *Wallet ID* 36502 el valor que corresponde es 3336353032.
	5A	*Account ID*	var. up to 10	cn	M	Identificador de una cuenta de una billetera en el ecosistema de QR de transporte. El *Account ID* no contiene el *Wallet ID*. Ver Anexo Técnico: *Account ID*. Ejemplo: - 000005227956984905FF para el *Account ID* 000005227956984905. - 12345678 para el *Account ID* 12345678. - 123456789F para el *Account ID* 123456789.

Tag (Hexa)		Descripción	*Length* (bytes)	Tipo	Mandatorio/ Opcional	Observación
	80	*Wallet Key ID (WK ID*)	2	b	M	Identificador de la *Wallet Key* que usó la billetera para firmar la *Account Public Key*. Formato binario no signado. Ejemplo: para el *Wk ID* 2 el valor 0002. Valor máximo permitido por el formato FFFF (Wk ID 65535)
	81	*Account Public Key (APK)*	32	b	M	Llave que generó el dispositivo para QR Data. Ejemplo: A8F48003EC8B7B8CF88EC108803D697570B83EE4A1F28BC3EF80A8C3C96DF3F3
	82	*Account Public Key Expiration Datetime*	6	cn	M	Fecha formato BCD YYMMDDHHMMSS. La fecha se expresa en GMT-0 (UTC) Ejemplo: 250107164233
	83	*Signed Account Public Key*	64	b	M	Se firma en orden: Wallet ID + Account ID + *Account Public Key Expiration Datetime* + *QR TTL* + *Feature flags* + *Account Public Key*.
	84	*QR Valid From*	6	cn	M	Fechas de inicio de validez del QR, en formato BCD (YYMMDDHHMMSS) La fecha se expresa en GMT-0 (UTC) Ejemplo: 250107164233 (para la fecha 2025 ENE 7 16h 42m 33s)
	85	*QR TTL*	3	b	M	Tiempo en segundos durante el cual este código QR es válido a partir de la fecha de inicio de validez (61 84 "*QR Valid From*"), en segundos, en formato binario. Formato binario no signado. Ejemplo: 00005A (90 segundos). Valor máximo admitido para el formato FFFFFF 16.777.215 (194d;4h;20m;15s)
	86	*Signature Algorithm*	1	cn	M	Algoritmo de firmado de *QRData* valores del 01 al 99, según la Tabla de Algoritmos. Formato numérico comprimido.

Tag (Hexa)		Descripción	*Length* (bytes)	Tipo	Mandatorio/ Opcional	Observación
						Ejemplo: 01 a 99 en HEX
	87	*Feature flags*	1	b	O	Campo para manejar diferentes *flags* (*bitwise*) de acuerdo a la tabla de QR *Feature Flags*: *Default:* 00000000
	88	*Issuer ID*	5	b (an)	M	Entidad emisora de la cuenta según Nómina de entidades Bancarias BCRA. Aclaración: En los casos que el emisor de la cuenta sea una billetera Digital interoperable, aplica el código de BCRA de emisores. Encodeado en formato binario ASCI-HEX el valor es un código de BCRA de emisores. Encodeado en ASCII ISO-8859-1. Para esta versión es un binario que se interpreta como an. Ejemplo: para el *Wallet ID* 36502 el valor que corresponde es 3336353032.
	9F08	*Application Version Number*	2	b	M	Versión del formato del QR que estamos definiendo en este estándar, definido como un entero. De 0000 a FFFF. Formato binario no signado. Ejemplo: para la versión 2 (versión actual de este documento), el valor que corresponde es 0002. Para la versión 3, el valor que corresponde será 0003. Para la versión 11, el valor que corresponde será 000B. Para la versión 65535, el valor que corresponde será FFFF.
	63	*Specific Transparent Template*	xxx	var	O	-
	xxxx	*Wallet Internal Data*	xx	var	O	Información propia de la *wallet* y no utilizada por el validador
	99	*Signed QR Data*	64	b	M	Firma de todos los campos ordenados del *Application Template* (61) excepto el *Signed QR Data*.

6.2. Seguridad

6.2.1. Integridad del código QR

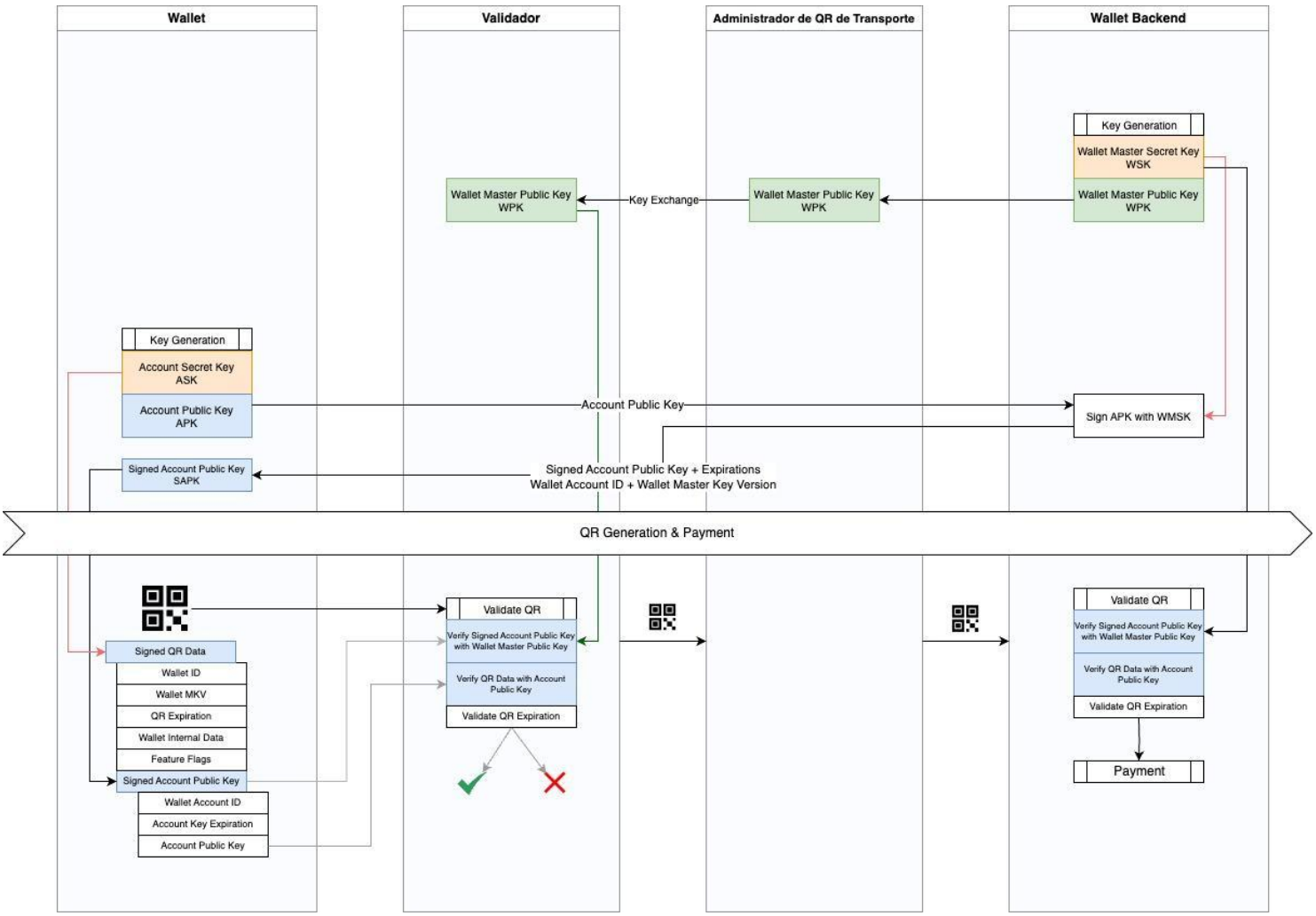
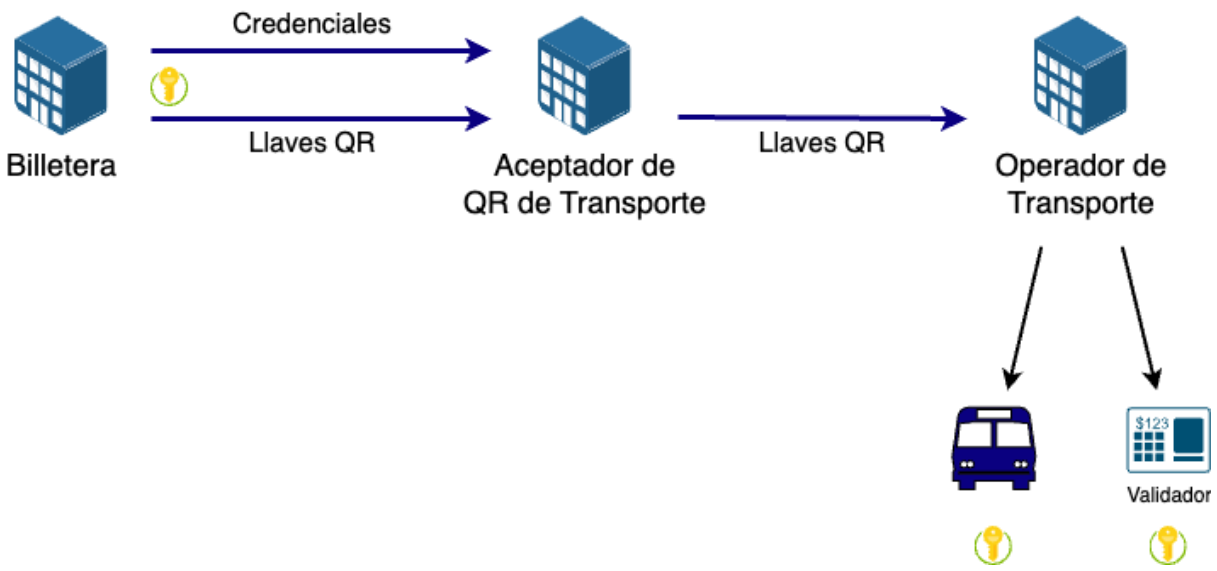


Diagrama de intercambio y uso de material criptográfico

La solución contempla un esquema de validación de integridad de los códigos QR en donde:

- La billetera firma los QR generados por sus usuarios y establece parámetros de caducidad de ese QR y/o firma. El firmado de los códigos QR se puede realizar aún cuando los usuarios no tengan conexión permanente a internet, permitiendo que puedan continuar generando códigos QR válidos en contextos de mala/nula conectividad.
- Los validadores, operadores de transporte y administradores QR pueden validar que los códigos QR que ingresan al sistema fueron firmados correctamente por la billetera.



La puesta en marcha del esquema de seguridad requiere un intercambio de llaves entre la billetera y el administrador. La billetera debe generar este par de llaves públicas y privadas (denominadas *Wallet Keys*) utilizando el algoritmo basado en curva elíptica *ED25519*.

La parte pública de esta llave debe ser compartida con el administrador QR, que será el encargado de suministrársela al operador del sistema de transporte (que la inyectará en los validadores).

Las *Wallet Keys* tendrán un identificador de versión que puede ser usado para la rotación de las llaves o su eventual cambio de algoritmo.

El dispositivo billetera del usuario generará un par de llaves pública y privada que denominamos *Account Keys*: *Account Secret Key* y *Account Public Key*. Periódicamente, cuando el dispositivo tenga conectividad, la *Account Public Key* se podrá regenerar y firmar en el *backend* de la billetera (ver validación código QR) con la *Wallet Secret Key*.

El validador, operador de transporte y administrador QR podrán validar que la *Account Public Key* es válida verificando su firma utilizando la *Wallet Public Key* almacenada en sus sistemas (que fue previamente aprovisionada en el proceso de configuración inicial).

Una vez validada la autenticidad de la *Account Public Key* se la utiliza para validar que el resto de los datos del QR no fueron adulterados. Estos datos se firman utilizando la *Account Secret Key*.

6.2.2. Material criptográfico

Material	Tipo	Propósito y características
Wallet Secret Key	*Wallet Key*	Usada en el *backend* de la billetera para firmar *Account Public Key*. Generada por la billetera.
Wallet Public Key	*Wallet Key*	Usada en el validador, operador de transporte, administrador QR y billetera para validar la firma de una *Account Public Key*. Generada por la billetera.
Account Secret Key	*Account Key*	Usada por el dispositivo del usuario para firmar datos de un código QR. Se renueva con periodicidad o cada vez que el dispositivo tenga conectividad. Generada por el dispositivo del usuario.
Account Public Key	*Account Key*	Usada en el código QR, el validador, operador de transporte, administrador de QR de transporte y billetera para validar la integridad de un código QR. Se renueva con periodicidad o cada vez que el dispositivo tenga conectividad. Generada por el dispositivo del usuario.
Signed Account Public Key	*Signed key*	Usada en el código QR, el validador, operador de transporte, administrador de QR de transporte y billetera para validar una *Account Public Key*. Es la *Account Public Key* firmada con la *Wallet Secret Key* y otros datos. El orden y datos es el siguiente: *Wallet ID* + *Account ID* + *Account Public Key Expiration Datetime* + *QR TTL* + *Feature flags* + *Account Public Key*. Generada en el *backend* de la billetera y retornada al dispositivo del usuario.
Signed QR Data	*Signed data*	Usada en el código QR para firmar datos del código QR. Usada por el validador, operador de transporte, administrador de QR de transporte y billetera para validar la integridad de los datos del código QR. Generada por el dispositivo del usuario.

6.2.3. Firmado de datos

Para firmar la *Account Public Key* junto con el *Wallet ID*, *Account ID*, expiración de la *Account Public Key*, *TTL* de cada QR y *Feature Flags*, el *backend* de la billetera realiza los siguientes pasos:

1) Se genera la *Wallet Account ID* que identifica al usuario en el sistema, concatenando el contenido de los *tags* 4F (*Wallet ID*) y 5A (*Account ID*).

Nota: Los valores utilizados serán consistentes con los generados en el *TLV* del QR (respetando *encoding*, longitudes y *padding*).

2) Se genera la fecha de expiración en bytes con la representación del tipo “cn” en base al EMV.

3) Se concatenan los bytes de *Wallet Account ID* + *Account Public Key Expiration Datetime* + QR *TTL* + *Feature flags* + *Account Public Key*.

4) Usando la *Wallet Secret Key*, se firma utilizando el algoritmo basado en curva elíptica *ED25519*.

5) Como resultado se obtiene el *Signed Account Public Key* (*tag* 83).

Para firmar los datos del QR, la billetera realiza los siguientes pasos:

1) Se toman los bytes completos (*tag* + *length* + *value*) de todos los *subtags* del *application template* (*tag* 61), excepto por el *tag* 99 que se estaría construyendo ahora: *Signable Data*.

2) Usando la *Account Secret Key*, se firma la *Signable Data* (punto 1) utilizando el algoritmo definido por la billetera (*tag* 86).

3) Como resultado de la firma se obtiene el *Signed QR Data* (tag 99).

6.2.4. Validación código QR

Para validar el QR se deben realizar los siguientes pasos:

1) Validar de que la trama del QR esté en Base64.

2) Validar que el *tag* 85 sea igual a CPV01.

3) Buscar el *application template* (*tag* 61) que incluya el *tag* 4F de una billetera válida y que esa billetera no esté en la lista de denegación de billeteras.

4) Buscar el ID de billetera (61 4F) para validar que hay una *Wallet Key ID* (61 80) en el validador.

5) Verificar que el QR haya sido firmado por la *Account Secret Key*:

    a) Se toman los bytes completos de los datos/*payload* del *application template* (*tag* 61).

b) De estos bytes, se elimina la *Signed QR Data* (*tag* 99) obteniendo los datos a ser verificados por la firma: *Signable Data*.

c) Utilizando el algoritmo definido en el *tag* 86 y la *Account Public Key* (*tag* 81), se verifica que la *Signed QR Data* (*tag* 99) se generó firmando la *Signable Data* con la *Account Secret Key*.

6) Verificar que *Account Public Key* (*tag* 81) haya sido firmada por la *Wallet Secret Key*. Para esto se utilizará la Wallet *Public Key* previamente compartida.

a) Se concatenan los bytes (sólo *value*) de *Wallet ID* (*tag* 4F) + *Account ID* (*tag* 5A), *Account Public Key Expiration Datetime* (*tag* 82) + QR *TTL* + Feature Flags y *Account Public Key* (*tag* 81), generando *Signable Data*.

b) Utilizando el algoritmo definido en el *tag* 86 *(Signature Algorithm) y la Wallet Public Key*, se verifica que la *Signed Account Public Key* (*tag* 83) se generó firmando la *Signable Data* con la *Wallet Secret Key*.

c) Se verifica que la *Account Public Key* no esté expirada utilizando *Account Public Key Expiration Datetime* (*tag* 82).

7) Verificar los *Feature Flags* (*tag* 87) retornados por el *backend* en el firmado de la *Account Public Key* para validar si pueden ser usados o no en este QR.

8) Verificar que el QR sea soportado para transporte utilizando el campo de *Feature Flags* (*tag* 87), específicamente la funcionalidad *Deny for transit* (ver Tabla de *QR Feature Flags*). *No aplica si el validador está en modo devolución.*

9) Validar que el QR no haya sido utilizado anteriormente en el mismo validador.

10) *Deny list*: verificar en *Feature Flags* (87) si la funcionalidad *Bypass deny list* (ver Tabla de *QR Feature Flags*) está activada en el QR. Luego validar *Wallet Account ID* en la Lista de denegación. *No aplica si el validador está en modo devolución.*

11) Validar que el QR esté dentro del tiempo de vida del QR definido por los campos *QR Valid* From (84) y *QR TTL* (85). El valor del campo *QR Valid From* tiene un *timestamp* (E.g. 250107164233) y el campo *QR TTL* tiene un valor en segundos (90). El código QR es válido en el periodo entre la *QR Valid From* y la suma de los segundos establecidos en *QR TTL* al momento establecido en *QR Valid From*. Por ejemplo para un *QR Valid From* 250107164200 y un QR TLL de 90 (segundos), el QR será válido entre 2025/01/07 16:42:00 y las 2025/01/07 16:43:30.

12) Validar *Wallet Account IDs* acumuladas por usuario con distintos QR en el mismo validador por la respuesta `*REJECTED_ACCOUNT_MAX_ATTEMPTS*`. Ver Parámetros de Riesgo, la sugerencia 5 pasadas por usuario en un lapso de 15 minutos.

6.2.5. Ejemplo de QR

6.2.5.1. Trama en Base64

hQVDUFYwMWGCAT9PBTM2NTAyWgoAAAUieVaYSQX/gAIAAYEgB/JDZCM6Fy+oGNX6e1CQC3rPc5XcG/CHY09qR
RYUmeiCBiUEJCNYAYNAcFTBv/BRqQ6dEsu/NoFHHO0g6UgQzKTaLxuAKh/HP/gUXnWjEbWIuouB+nhMvTERnhgZve
PaCfSd4TYsgk4RDYQGJQQjFDAAhQMAAFqGAQGHAQKIBTM2NTAynwgCAAJjWQFXaFEwNTUwMDUzNjg4NDM4Mz
k5NjE0MTEjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMwMyMjLTM0LjU3NTEyMSwtNTguNDM0NzIzmUA
wxFOSO9g247DWrUXVNw81aY9xBlbfPAUYtMHmVmrm5MGNA73dUzXdsnqJ5bruBPbwKwV7OH1bkJyWwVb4Ok0K

6.2.5.2. Trama en HEX

850543505630316182013F4F0533363530325A0A00000522795698490 5FF80020001812007F24364233A172FA818
D5FA7B50900B7ACF7395DC1BF087634F6A45161499E882062504242358018340705 4C1BFF051A90E9D12CBBF36
81471CED20E94810CCA4DA2F1B802A1FC73FF8145E75A311B588BA8B81FA784CBD31119E1819BDE3DA09F49DE
1362C824E110D840625042314300085030000 5A8601018701028805333635303 29F0802000263590157685130353
530303533363838343338333939363134313123232323232323232323232323232323232323232323232323232
323232323232323232323232303323232D33342E3537353132312C2D35382E343334373233994030C453923BD836
E3B0D6AD45D5370F35698F710656DF3C0518B4C1E6566AE6E4C18D03BDDD5335DDB27A89E5BAEE04F6F02B057
B387D5B909C96C156F83A4D0A

6.2.5.3. Trama en HEX formateada

85 05 4350563031
61 82 01 3F
 4F 05 3336353032
 5A 0A 00000522795698490 5FF
 80 02 0001
 81 20 07F24364233A172FA818D5FA7B50900B7ACF7395DC1BF087634F6A45161499E8
 82 06 250424235801
 83 40
7054C1BFF051A90E9D12CBBF3681471CED20E94810CCA4DA2F1B802A1FC73FF8145E75A311B588BA8B81FA784
CBD31119E1819BDE3DA09F49DE1362C824E110D
 84 06 250423143000
 85 03 00005A
 86 01 01
 87 01 02
 88 05 3336353032

9F08 02 0002
63 59
01576851303535303035333638383433383339393631343131232323232323232323232323232323232323232323232323232323232323232323232323232323232323232323232323232323232323232323232323232323232323303323232D33342E3537353132312C2D35382E343334373233
99 40
30C453923BD836E3B0D6AD45D5370F35698F710656DF3C0518B4C1E6566AE6E4C18D03BDDD5335DDB27A89E5BAEE04F6F02B057B387D5B909C96C156F83A4D0A

6.2.5.4. Ejemplo de validación de *Signed Account Public Key* sobre el QR:

Wallet ID: 3336353032

Account ID: 00000522795698490 5FF

Wallet Account ID: 33363530320000052279569849 05FF

APK Expiration Datetime: 250424235801

Account Public Key:
07F24364233A172FA818D5FA7B50900B7ACF7395DC1BF087634F6A45161499E8

Signable Data (*Wallet Account ID* + *APK Expiration Datetime* + *QR TTL* + *FeatureFlags* + *APK*):
33363530320000052279569849 05FF2504242358010000 5A0207F24364233A172FA818D5FA7B50900B7ACF7395DC1BF087634F6A45161499E8

Signed Account Public Key:
7054C1BFF051A90E9D12CBBF3681471CED20E94810CCA4DA2F1B802A1FC73FF8145E75A311B588BA8B81FA784CBD31119E1819BDE3DA09F49DE1362C824E110D

Wallet Public Key:
D3F6A94215115CCE52443D6FE2622C3C6ACC560A9697E161513B7FE91A47D111

6.2.5.5. Ejemplo de validación de *Signed QR Data* sobre el QR:

Signable Data (contenido del *Application template* menos el *tag* 99):
4F0533363530325A0A0000052279569849 05FF80020001812007F24364233A172FA818D5FA7B50900B7ACF7395DC1BF087634F6A45161499E882062504242358018340705 4C1BFF051A90E9D12CBBF3681471CED20E94810CCA4DA2F1B802A1FC73FF8145E75A311B588BA8B81FA784CBD31119E1819BDE3DA09F49DE1362C824E110D840625042314300085030000 5A8601018701028805333635303 29F0802000263590157685130353530303533363838343338333939363134313123232323232323232323232323232323232323232323232323232323232323232323232323232323232323232323232323232323232323232323232323232323303323232D33342E3537353132312C2D35382E343334373233

Account Public Key:
07F24364233A172FA818D5FA7B50900B7ACF7395DC1BF087634F6A45161499E8

Signed QR Data:
30C453923BD836E3B0D6AD45D5370F35698F710656DF3C0518B4C1E6566AE6E4C18D03BDD
D5335DDB27A89E5BAEE04F6F02B057B387D5B909C96C156F83A4D0A

6.2.6. Autorización de los servicios

Los servicios HTTP que provee la billetera y administrador QR serán autenticados mediante la utilización de *OAuth2* con la provisión segura de un *client_id* y *client_secret*.

6.3. Servicios de procesamiento

La billetera desarrollará un servicio para responder la solicitud de viajes. Dicho servicio deberá correr los mecanismos de seguridad correspondientes y validar los fondos de cada usuario. Deberá contar también, con un servicio de devoluciones que permita retornar fondos en caso de que exista interrupción del servicio de transporte.

La API está definida según el documento *Swagger* QR transporte AR.

6.4. Servicios de denegación

El administrador QR implementará un servicio para la gestión de las listas de denegación en una "API de denegación". Dicho servicio permitirá a las billeteras agregar o quitar usuarios de las listas de denegación presentes en los validadores, servicios del operador de transporte, el administrador QR y las billeteras.

El servicio de denegación también debe soportar denegar en el validador a una billetera. Una billetera podría solicitar al administrador denegar los viajes en caso de contingencia.

La API está definida según el documento *Swagger* QR transporte AR.

6.5. Servicios de Intercambio de Claves (*Keystore*)

La billetera implementará un servicio denominado "*Keystore*" donde deja disponibles las claves necesarias para la configuración inicial.

El *Keystore* almacena llaves de tipo *Wallet Keys* (ver Material Criptográfico) que son públicas. Estas llaves se utilizan para validar la autenticidad de los códigos QR generados por los dispositivos.

Las *Wallet Keys* tienen los siguientes atributos:

- *id*: identificador de la clave para una billetera, *string* numérico de 4 caracteres (e.g.: 0004)
- *wallet_id*: identificador numérico para una una billetera, *string* numérico (e.g.: 36558)
- *wallet_public_key*: la parte pública de la clave *Wallet Key*.
- *valid_from*: momento desde el cual una clave es válida, *string ISO date* (e.g.*:* 2000-10-31T01:30:00-00:00).
- *valid_to*: momento hasta el cual una clave es válida, *string ISO date* (e.g.: 2000-10-31T01:30:00-00:00).
- *status*: estado de la clave, *string* con los valores *active* o *inactive*.
- *signature_algorithm*: algoritmo que va a utilizar esta llave, *string* con los valores [ED25519].
- *created_at*: momento en el que se creó la clave, *string ISO date* (e.g.: 2000-10-31T01:30:00-00:00).
- *updated_at*: momento en el que se actualizó la clave por cambio de estado, string ISO date (e.g.: 2000-10-31T01:30:00-00:00).

6.5.1. *Status active*

Las *Wallet Keys* en estado *active* deben ser propagadas e inyectadas a los validadores por el administrador QR y operador de transporte.

6.5.2. *Status inactive*

La billetera puede cambiar una llave al estado *inactive*. Este estado es final y no puede ser revertido. El cambio de estado debe estar reflejado en el campo *updated_at* definido previamente. Cuando una *Wallet Key* pasa a estado *inactive* deben ser invalidadas y removidas de los validadores, denegando todo viaje que se intente realizar con códigos QR generados con las mismas.

6.5.3. Actualización de las llaves

El administrador QR consultará periódicamente la existencia de nuevas *Wallet Keys* e implementará un servicio de actualización para que la billetera pueda forzar una actualización de claves de forma proactiva.

Las API de los servicios están definidas según el documento *Swagger* QR transporte AR.

6.6. Servicio de reportes

El administrador QR proporcionará un servicio para que el operador de transporte y la billetera puedan conectarse para intercambiar reportes. El servicio será del tipo *SFTP*.

6.7. *Account IDs*

Una *Account ID* representa a un usuario o un medio de pago del usuario en el sistema. Las billeteras asignan un *Account ID* a cada uno de sus usuarios.

Ejemplos de representaciones del *Account ID*:

Valor en el QR	**Account ID**	**Comentario**
12 34 56 78	12345678	*Account ID* tiene una cantidad de números par.
12 34 56 78 9F	123456789	*Account ID* tiene una cantidad de números impar, se hace *padding* de F para completar hasta tener una cantidad de caracteres par.
00 00 06 78 90	0000067890	*Account ID* con 0 (ceros) a la izquierda, cantidad de números par.
00 00 05 22 79 56 98 49 05 FF	000005227956984905	*Account ID* con `0` (ceros) a la izquierda con una cantidad de números impar. El estándar *EMVCo* lo soporta, no incentivamos el *padding* de FF a la derecha.

6.7.1. *Wallet Account ID*

Los administradores QR utilizan este *Account ID* junto con el *Wallet ID* (llamado "Wallet Account ID") para:

- Conformar una clave e identificar a usuarios de las billeteras en los servicios de Denegación.
- Conformar una clave e identificar a usuarios de las billeteras en validadores para denegar viajes.
- Conformar una clave y validar la integridad de los códigos QR durante el firmado de datos.

Ejemplo:

```
4F 05 3336353032
5A 05 123456780F
```

- *Wallet ID* (4F): 36502
- *Account ID* (5A): 123456780
- *Wallet Account ID*: 36502123456780

6.8. Tabla de configuraciones

6.8.1. Tabla de *QR Feature Flags*

Funcionalidades del campo 87 del Código QR:

Posición	**Valor**	***Feature***	**ID**	**Propósito y características**
1	0b00000001	*Deny for transit*	*deny_for_transit*	QR no permitido para su uso en el sistema de transporte. Si este *feature* está encendido, este QR no es válido para el sistema de transporte independientemente del resto de los *feature flags*. El validador deberá rechazarlo y notificar este evento.
2	0b00000010	*Bypass deny list*	*bypass_deny_list*	No validar la existencia en la lista de denegación. En caso de existir deuda, esta se transfiere al emisor/billetera.

6.8.2. Tabla de Algoritmos

Algoritmos definidos para el campo 86 del Código QR:

Valor	**Algoritmo**	**Propósito y características**
01	ED25519	Algoritmo basado en ED25519.

6.9. Parámetros de Riesgo

https://docs.google.com/spreadsheets/d/1i6nJXS76pCWVBa0T9AvA5JpGHhRkb8bPH9whrf2FWTO8/edit?gid=1202513083#gid=1202513083

ID Parámetro	Componente	Parámetro	Valor
1	*Wallet*	Expiración de las llaves para firmar los QR (tiempo máximo que el usuario puede estar sin conectividad).	7 días (max)
2	*Wallet*	Expiración de cada QR (*valid from* & *valid to*).	90 segundos (max) Luego de este tiempo es riesgo Emisor
3	*Wallet*	*Refresh rate QR* en la *Wallet* (cada cuanto tiempo la *Wallet* "refresca" el QR en pantalla).	3s (min) 8s (max)
4	Validador	*Refresh rate denylist*: tiempo entre que la billetera le indica que una cuenta debe ser agregada a la *denylist* y se refleja en los validadores.	Near real time 2h (max)
5	Validador	Cuanto tiempo sin conexión va a estar el validador antes de rechazar QRs (definido idem a las marcas).	2h (max)
6	Validador	Cuántas veces acepta en un mismo validador el mismo QR.	1
7	Validador	Cuantas acumuladas por usuario en el mismo validador, incluso con distintos QRs en cuanto tiempo.	5 viajes en 15 minutos
8	Validador	*Expiration* de cada QR (*valid from* & *valid to*)	90 seg
9	*Payment*	Delta entre *scan* y el pago.	1 mes (max)
10	*Payment*	Cuántas veces se puede pagar con el mismo QR.	1 en el mismo validador
11	*Payment*	Tarifa máxima del sistema.	50000
12	Validador	*TTL* (*Time to limit*) de la lista de denegación.	7 días
13	Administrador	Cantidad de días y veces de reintentos para recupero de deuda.	3 x día, durante 15 días
20	*Payment*	Saldo mínimo para generar un QR.	1200
21		Cuando una billetera se cae re-intentamos durante *x* tiempo: **a definir por administradores* i) Se cae riesgo operador ii) Se cae riesgo billetera	Se respeta último estado del usuario