Homologación de información de tokens entre banco y franquicias

Juan Fernando Lopera Muñoz

Trabajo de grado presentado para optar al título de Ingeniero de Sistemas

Asesores
Maria Bernarda Salazar Sánchez, Ph.D. en Ingeniería Electrónica
Santiago Cardona García, Líder de línea de conocimiento

Universidad de Antioquia
Facultad de ingeniería
Ingeniería de sistemas
Medellín
2025

Cita	(Lopera Muñoz, 2025)
Referencia **Estilo APA 7 (2020)**	Lopera Muñoz, J. F (2025). *Homologación de información de tokens entre banco y franquicia.* Trabajo de grado profesional, Ingeniería de sistemas, Universidad de Antioquia, Medellín, 2025

Dedicatoria

Para mi familia cuyo apoyo y acompañamiento fue innegable y fueron mi motor durante todo este proceso académico, también para mis amigos y todas aquellas personas que de alguna u otra manera hicieron parte de este lindo trayecto académico dejando aprendizajes, vivencias y experiencias que me marcaron de una buena manera permitiéndome crecer en el ámbito personal y laboral.

Agradecimientos

Deseo manifestar mi más profunda gratitud a Bancolombia, por haberme brindado la oportunidad de realizar mi práctica profesional. En especial, reconozco el acompañamiento incesante de mi mentor durante este trayecto Jader Atehortúa, y a mi líder Santiago Cardona, quienes me guiaron, facilitaron y apoyaron en cada etapa del proceso, así como el respaldo y la disposición de todo el equipo de Spartans.

Tabla de contenido

Lista de figuras

Siglas, acrónimos y abreviaturas

AWS	Amazon Web Services
CI/CD	Integración Continua/Despliegue Continuo
E2E	Pruebas End to End
QA	Quality Assurance o Garantía de Calidad

Resumen

Bancolombia cuenta actualmente con un módulo autorizador de digitalizaciones con el cual diversas franquicias como lo son MasterCard, VISA y American Express envían solicitudes de digitalización de tarjetas crédito y débito, en ocasiones, la comunicación entre las franquicias y el banco se ve interrumpida por lo que el banco queda con la información incompleta de las digitalizaciones de una tarjeta. A partir de esta falencia se propone generar un sistema que haga las respectivas consultas por el estado actual de los tokens a las distintas franquicias para así homologar la información entre banco y franquicias, especialmente para los que han sido eliminados o suspendidos, lo que corresponde a una necesidad del banco que consiste en tener un control sobre el total de tokens existentes y cuáles son sus estados actuales.

En este informe se podrá encontrar cuales son los objetivos que se buscan cumplir con el desarrollo de este proyecto, asimismo, cual es la metodología empleada y como el proyecto permitió al banco homologar la información de 200-500 tokens al día, generando al banco un control más eficiente de la cantidad de tokens en estado activo y de los gastos que pueden conllevar estos.

Palabras clave: Pagos digitales, digitalización de tarjetas, homologación de información, tarjeta de crédito, tarjeta de débito, eliminación, token

Abstract

Bancolombia currently has an authorization module for digitizations, which various franchises—such as MasterCard, VISA, and American Express—use to send digitization requests for credit and debit cards. Sometimes, communication between these franchises and the bank is interrupted, leaving the bank with incomplete digitization information for a card. To address this shortcoming, it is proposed to develop a system that queries the current status of tokens from the different franchises in order to standardize the information between the bank and the franchises, especially for those tokens that have been deleted or suspended. This meets the bank's need to maintain control over all existing tokens and their current statuses.

This report outlines the objectives to be achieved through the development of this project, as well as the methodology employed. It also explains how the project enabled the bank to standardize the information for 200-500 tokens per day, thereby allowing for more efficient control over the number of active tokens and better management of the expenses associated with them.

Keywords: Digital payments, card digitization, information homologation, credit card, debit card, token elimination, tokenization

1. Introducción

Algunas entidades bancarias cuentan con un módulo autorizador de digitalizaciones que recibe solicitudes de distintas franquicias para la digitalización de tarjetas de crédito y débito. Sin embargo, en ocasiones se producen interrupciones en la comunicación, lo que deja al banco con información incompleta acerca de las tarjetas digitalizadas. Por lo anterior, en este proyecto busca homologar la información de los estados de las digitalizaciones entre el banco y la franquicia, especialmente con tokens eliminados o suspendidos. A nivel metodológico se abordfa desde la contextualización del problema, el desarrollo de la solución validando su correcta implementación ejecutando pruebas que verifiquen el cumplimiento de los requisitos funcionales y no funcionales para posteriormente hacer un despliegue a entorno productivo hasta el monitoreo de forma continua para obtener la cantidad de tokens actualizados diariamente.

2. Objetivos

2.1 Objetivo general

Homologar la información entre la entidad bancaria y la franquicia de los estados de los tokens asociados a una tarjeta débito y crédito que han sido eliminados o suspendidos cuya actualización no fue satisfactoria retornando el estado actual de los tokens asociados.

2.2 Objetivos específicos

- Determinar el flujo que permita hallar los tokens que han sido eliminados y cuya información no ha sido compartida con el banco.
- Implementar el flujo que comunique al banco con las franquicias e intermediarios para obtener toda la información necesaria para obtener los tokens que han sido eliminados
- Validar por medio de pruebas del flujo los requisitos no funcionales en términos de la cantidad de información homologada entre el banco y las franquicias.

3. Marco teórico

En el marco de la cuarta revolución industrial, también conocida como industria 4.0, los avances tecnológicos han transformado radicalmente la forma en la que se interactúa con el mundo digital. Uno de los aspectos claves en esta transformación es la digitalización de servicios y productos, la cual juega un papel crucial en la modernización de los sistemas de pago.

La digitalización de tarjetas, también conocida como "tokenización", se puede definir como "el proceso que permite proteger datos sensibles, sustituyéndolos por equivalentes no-sensibles" (Damen, 2020), es decir, se genera un valor aleatorio que sustituye la información de la tarjeta de un usuario y este es conocido como "token". El token se puede utilizar en transacciones en línea sin exponer los datos originales de la tarjeta protegiendo así a los usuarios frente a posibles fraudes.

El proceso de "tokenización" en Bancolombia consta de las siguientes fases:

- **Fase 1.** Un usuario hace la solicitud de digitalización al ingresar los datos de su tarjeta ya sea de débito o crédito en una billetera electrónica, comercios electrónicos o dispositivos inteligentes
- **Fase 2.** La billetera, comercio o dispositivo envía la solicitud de digitalización a Bancolombia con la información de la tarjeta ingresada
- **Fase 3.** Bancolombia realiza las validaciones pertinentes para verificar la validez de la información, verifica que el cliente sea apto para digitalizar su tarjeta, es decir, que no tenga bloqueos en ellas y analiza que tan seguro es el dispositivo que hizo el envío de la solicitud. Con los resultados obtenidos se toma la decisión para saber si se aprueba o rechaza la solicitud de digitalización.
- **Fase 4.** Si la solicitud fue aprobada, se envía la solicitud a la franquicia correspondiente, actualmente Bancolombia opera con VISA, MasterCard y American Express
- **Fase 5.** La franquicia digitaliza (tokeniza) la tarjeta y el token es enviado a Bancolombia y a la billetera virtual, comercio o dispositivo que inició el proceso
- **Fase 6.** El usuario podrá empezar a realizar pagos digitales con sus dispositivos inteligentes o en comercios electrónicos sin necesidad de tener a mano su tarjeta.

Los “tokens” tienen un ciclo de vida, en el cuál pueden ser eliminados, suspendidos o desactivados. Hay momentos que por factores de fallos en la conexión entre franquicias y banco la información que se genera al realizar la eliminación de un token no llega completa o no llega en su totalidad al banco, por lo que se requiere hacer una búsqueda de esta información que le permitirá al banco tener toda la información y el control necesario sobre la cantidad de tokens activos, eliminados, suspendidos o inactivos de los clientes.

Este proceso, actualmente existe para homologar la información de las tokenizaciones que no quedaron completadas, es decir, la información generada en el proceso inicial de la tokenización no fue enviado por las franquicias, por lo que para el banco no es posible validar y conocer con exactitud cuál es el token que se generó y toda la información correspondiente a este. Además, dicho proceso cuenta con unas tecnologías de base lo suficientemente robustas como lo son Java con su framework de Spring, una base de datos relacional como PostgreSQL y se encuentra desplegado en la nube con el proveedor de servicios de AWS. Además, este sistema ha sido diseñado con prácticas de CI/CD y DevOps, lo que permite una implementación ágil y eficiente de nuevas funcionalidades, así como una mayor confiabilidad en la gestión del ciclo de vida del software. Estas características aseguran que cualquier mejora o modificación en el proceso pueda integrarse de manera continua y controlada, reduciendo el riesgo de fallos en producción y garantizando la calidad del servicio.

4. Metodología

Para lograr la correcta implementación del proyecto de la actualización de la eliminación de token asociado a una tarjeta se tuvo las siguientes fases:

4.1 Contextualización del proceso de digitalización de tarjetas

Para iniciar se buscó comprender a fondo el proceso actual de digitalización de tarjetas dentro de Bancolombia y el proceso que se ejecuta para obtener la información de los tokens eliminados cuya actualización no fue enviada de forma correcta por parte de la franquicia. El objetivo era obtener una visión clara de cómo funciona el flujo operativo actual, identificar las herramientas y tecnologías involucradas, y analizar en detalle el proceso que se implementó y el cual generó una conexión óptima y eficaz con las distintas franquicias que permitió actualizar la información de los distintos tokens asociados a una tarjeta de débito o crédito de un cliente.

4.2 Desarrollo de la propuesta

Después de entender cómo funciona, se procedió a la implementación del nuevo proceso para obtener los tokens eliminados y actualizar la información en la base de datos del banco. En este punto se realizó el análisis de requisitos funcionales y no funcionales para ver cuál era el objetivo final del desarrollo y enfocar el desarrollo en el cumplimiento de estos.

Los requisitos funcionales que “se refieren lo que el sistema debe hacer” (Sommerville, 2011), se definieron de la siguiente manera:

- El sistema debe actualizar el estado de los tokens cuyo estado actual en la base de datos del módulo autorizador sea activo o suspendido y para las franquicias sea eliminado
- El sistema debe consultar aquellos tokens que lleven exactamente un mes desde su creación.
- El sistema debe hacer las consultas en las distintas franquicias haciendo diferenciación por cada una de las billeteras electrónicas.

Por otra parte, los requisitos no funcionales que “se relacionan directamente con los servicios específicos que el sistema entrega a sus usuarios. Pueden relacionarse con propiedades

emergentes del sistema, como fiabilidad, tiempo de respuesta y uso de almacenamiento" (Sommerville, 2011), se definieron como se observa a continuación:

- Máximo se deben enviar 1500 registros a las franquicias para actualizar
- Se espera un crecimiento de un 20% de los registros a enviar en los próximos 2 años
- El proceso de actualización se debe finalizar como máximo en 1 hora
- El porcentaje máximo de error es de 10%.

El módulo ya contaba con una base desarrollada en Spring Boot, PostgreSQL y AWS, la cual se adaptó y se configuró para cumplir con los requisitos definidos anteriormente. En esta fase se hicieron pruebas manuales en cada uno los distintos entornos de desarrollo y producción, para asegurarse de que siempre se cumplan con los requisitos funcionales planteados.

4.3 Ejecución de pruebas

Se ejecutaron pruebas exhaustivas en entornos controlados para validar la efectividad de la nueva implementación y el cumplimiento tanto de los requisitos funcionales como los no funcionales. Estas incluían simulaciones de escenarios diversos, tanto normales como críticos, para medir la fiabilidad, el desempeño, la precisión y la seguridad del sistema. Dichas pruebas se realizaron en un entorno de QA.

Para la validación de los requisitos funcionales, se implementaron pruebas unitarias cuyo objetivo es que se diseñan "para verificar que el bloque de código se ejecuta según lo esperado, de acuerdo con la lógica teórica del desarrollador" (AWS, s/f). Asimismo, se llevaron a cabo pruebas de aceptación en las que se "verifica si todo el sistema funciona según lo previsto" (IBM, 2024), y las pruebas E2E, en las que se "verifica todo el flujo de una aplicación desde el inicio hasta el final, simulando el comportamiento del usuario en un escenario real" (Cabaña, 2023).

Por otra parte, para realizar la validación de los requisitos no funcionales se ejecutaron distintas pruebas de performance, las cuales son pruebas de estrés en las que "se pone a prueba la robustez y la confiabilidad del producto, sometiéndose a condiciones de uso extremas" (Congote & Hincapie, 2020) y pruebas de carga, las cuales tienen como objetivo "determinar cuáles son los

flujos más críticos para implementar una mejora en los mismos y detectar posibles cuellos de botella para corregirlos y así, optimizar el rendimiento" (Congote & Hincapie, 2020).

4.4 Paso a producción

Una vez que las pruebas de la fase anterior arrojan resultados satisfactorios y confirmaron que se cumplieron todos los requisitos, se avanza a la siguiente etapa, donde se realizó el despliegue en un entorno productivo en el que se inició con la búsqueda de los tokens que han sido eliminados y su posterior actualización en la base de datos del banco.

4.5 Estabilización del producto

Por último, una vez que el sistema fue desplegado en un entorno productivo, se llevó a cabo un monitoreo continuo con el propósito de asegurar su fiabilidad, estabilidad y rendimiento. De esta forma, se podrán identificar a tiempo eventuales anomalías y aplicar las medidas correctivas o preventivas necesarias.

4.6 Cronograma de actividades

En la Figura 1 se muestra el cronograma establecido, donde es posible apreciar con mayor detalle la duración de cada una de las fases mencionadas. Cabe resaltar que en la organización se utiliza Scrum, este se define como "un marco de trabajo liviano que ayuda a las personas, equipos y organizaciones a generar valor a través de soluciones adaptativas para problemas complejos." (Schwaber & Sutherland, 2020). Por este motivo, la duración de cada fase no solo se basa en fechas, sino también en Sprints, los que consisten en iteraciones de tiempo fijas, usualmente entre 2 semanas y un mes, en las que se planifican y completan tareas concretas (Schwaber & Sutherland, 2020), durante estas iteraciones de tiempo se irán haciendo mejoras y completando cada una de las fases mencionadas anteriormente.

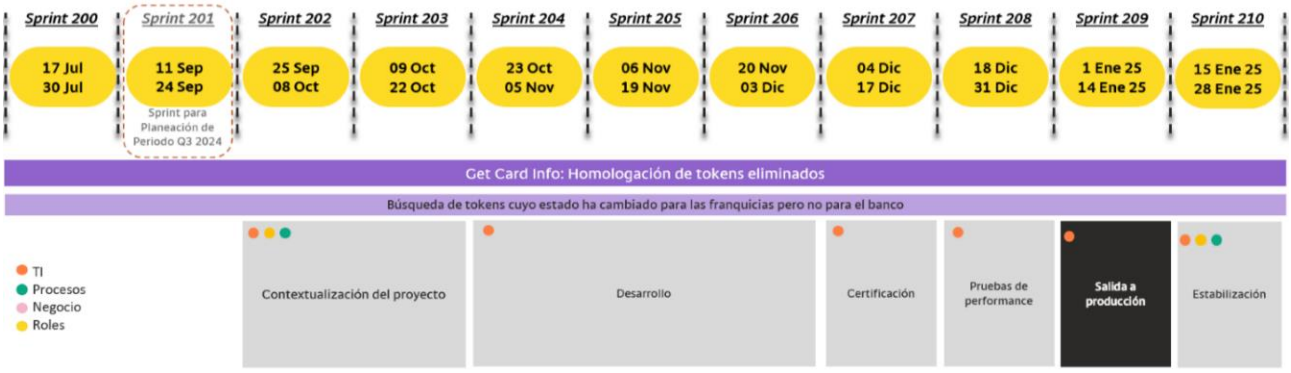

Figura 1 Cronograma de actividades

5. Resultados

Al analizar el flujo existente se encontró una oportunidad de mejora que consistió en unificar y optimizar la lógica existente, para ello se llevó a cabo una refactorización aplicando el patrón de diseño Template Method, cuya premisa principal es “definir un esqueleto común para un algoritmo y delegar los detalles de implementación a las clases hijas” (Parra, 2024a). Esta mejora permitió integrar en un solo flujo dos procesos, el que ya existía para completar digitalizaciones que no fueron completadas y el implementado para actualizar los estados de tokens eliminados y suspendidos que, aunque similares en su funcionamiento general, requerían acciones específicas diferentes según las necesidades de cada caso.

Para llevarlo a la práctica, se creó una clase abstracta que define los métodos abstractos y concretos que conforman la estructura general del flujo (ver Figura 2). Los métodos abstractos deben ser implementados por las clases hijas, permitiendo adaptar únicamente las partes que varían sin alterar la lógica principal. Al mismo tiempo, los métodos concretos proporcionan comportamientos comunes y reutilizables para todas las clases que extienden de esta clase base. Como resultado, los dos procesos se integraron en un único esquema de ejecución, lo que permitió compartir secciones comunes y reducir la duplicación de código de forma significativa. Al aislar los detalles de implementación en clases hijas, se facilitó tanto la ampliación como el mantenimiento de la solución, pues cada proceso puede personalizarse sin alterar la lógica general. En la Figura 2 puede observarse tanto la definición de la clase abstracta como las clases concretas que heredan de ella, ilustrando cómo cada proceso específico implementa o extiende los métodos necesarios para ajustarse a sus propios requerimientos.

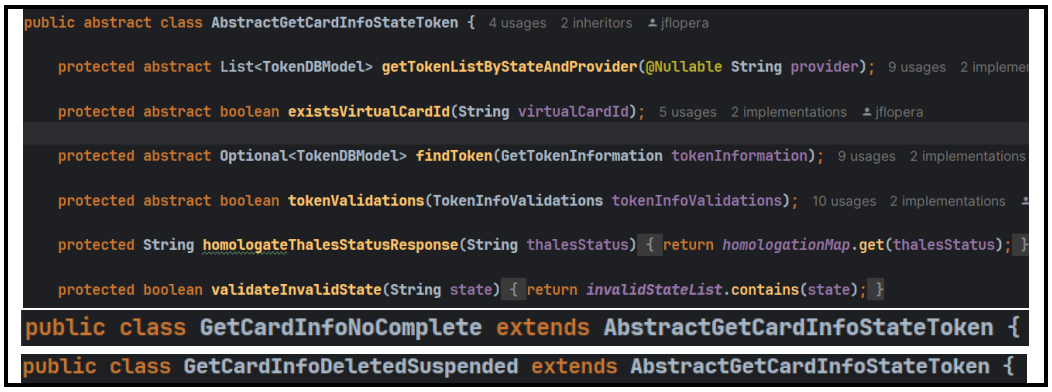

Figura 2 Implementación del patrón Template Method

Siguiendo la misma línea de aplicar patrones de diseño adecuados, se recurrió al uso del patrón Strategy para gestionar de forma flexible los flujos de ejecución según la acción que se desee llevar a cabo. Este patrón “permite definir una familia de algoritmos o clases de manera que sean intercambiables entre sí, y que el consumidor de estos no sepa si está utilizando el algoritmo 1, el 2 o cualquier otro” (Parra, 2024b).

Gracias a esta implementación, el sistema puede cambiar el flujo de ejecución en tiempo real, sin necesidad de determinar de antemano cuál de los algoritmos específicos se va a utilizar ni de reiniciar el sistema. De este modo, se logra mayor flexibilidad y escalabilidad, pues cada estrategia se encarga de un comportamiento particular y puede agregarse, modificarse o intercambiarse sin alterar la lógica central. En la Figura 3 se muestra cómo se llevó a cabo la implementación, mostrando el uso de las distintas clases concretas y la forma en que estas se conectan con el resto del sistema, lo que permitió la elección dinámica del flujo requerido para cada situación en tiempo de ejecución.

```
public class GetCardInfoServiceImpl implements GetCardInfoService{

    private final GetCardInfoNoComplete getCardInfoNoComplete;  2 usages
    private final GetCardInfoDeletedSuspended getCardInfoDeletedSuspended;  2 usages

    public GetCardInfoServiceImpl(GetCardInfoNoComplete getCardInfoNoComplete,  1 usage  jflopera
                                  GetCardInfoDeletedSuspended getCardInfoDeletedSuspended){...}

    public void execute(@NotNull GetCardInfoRequest getCardInfoRequest) {  4 usages  jflopera +1
        AbstractGetCardInfoStateToken getCardInfo = switch (getCardInfoRequest.getState()) {
            case Constants.NOT_COMPLETE -> getCardInfoNoComplete;
            case Constants.DELETED_SUSPENDED -> getCardInfoDeletedSuspended;
            default -> throw new ForbiddenDataException(Constants.ERROR_STATE_BODY);
        };
        List<TokenDBModel> list = getCardInfo.initialize(getCardInfoRequest);
        getCardInfo.execute(list);
    }
```

Figura 3 Implementación del patrón Strategy

Para llevar a un entorno productivo la implementación realizada, se requiere pasar por distintas fases definidas en un pipeline (definición y referencia), dicho pipeline puede ser observado en la Figura 4, en la que se definen los despliegue a los distintos entornos de desarrollo y calidad, donde se realizaron pruebas de aceptación para el entorno de desarrollo y la revisión de la calidad de código en la plataforma de SonarCloud junto con el análisis de vulnerabilidades y en el entorno

de calidad se hicieron pruebas de aceptación, de performance y pruebas E2E, además de revisión de código estático todas las pruebas mencionadas fueron mencionadas en la metodología

Para llevar la implementación a un entorno productivo, resulta fundamental contar con un pipeline de despliegue que establezca las fases y controles de calidad necesarios, tal como se ilustra en la Figura 4. Este pipeline define los pasos y validaciones previas que garantizan que las modificaciones en el código cumplan con los requisitos de funcionalidad, rendimiento y seguridad antes de llegar a producción.

En primer lugar, se realizó el despliegue en el entorno de desarrollo, en este, se llevaron a cabo pruebas de aceptación y se analiza la calidad del código mediante análisis de código estático con la plataforma SonarCloud con el fin de identificar problemas de estilo o complejidad desde etapas tempranas y además se evaluaron vulnerabilidades en la plataforma Fluid Attacks para conservar altos niveles de seguridad y mantenibilidad. Posterior a esto se realiza el despliegue al entorno de calidad en el cual se aplican pruebas más rigurosas: pruebas de aceptación, pruebas de performance y pruebas E2E, dichas pruebas fueron descritas en el apartado de metodología. Además, en el entorno de calidad, también se realizaron revisiones de código estático y de vulnerabilidades.

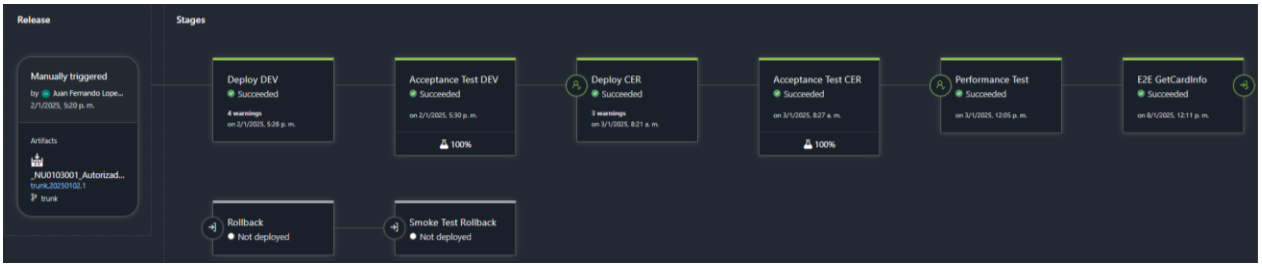

Figura 4 Pipeline de despliegue

El funcionamiento del proceso implementado comienza identificando las digitalizaciones o “tokenizaciones” que han cumplido exactamente un mes desde su activación, como se ilustra en la Figura 5. Para ello, el sistema revisa la fecha de activación de cada registro (columna “date_request”) y determina automáticamente cuáles requieren verificación en ese momento.

	pk_id_visa_request	issuer_card_ref_id	virtual_card_id	state_token	date_request	date_update
1	2.513.248	issuer_gtk_del_sus_apple_1	virtual_card_gtk_del_sus_apple_1	ACTIVATE	2024-12-13 09:15:13.640	[NULL]
2	2.513.249	issuer_gtk_del_sus_apple_2	virtual_card_gtk_del_sus_apple_2	ACTIVATE	2024-12-13 09:15:13.640	[NULL]
3	2.513.250	issuer_gtk_del_sus_apple_3	virtual_card_gtk_del_sus_apple_3	ACTIVATE	2024-12-13 09:15:13.640	[NULL]
4	2.513.251	issuer_gtk_del_sus_apple_4	virtual_card_gtk_del_sus_apple_4	SUSPEND	2024-12-13 09:15:13.640	[NULL]
5	2.513.252	issuer_gtk_del_sus_apple_5	virtual_card_gtk_del_sus_apple_5	SUSPEND	2024-12-13 09:15:13.640	[NULL]

Figura 5 Datos antes del inicio del proceso de actualización

Una vez definidos los registros a validar, el sistema envía el valor de la columna "issuer_card_ref_id", cuyo valor es asociado al número de tarjeta del cliente, a las franquicias (ver Figura 6), las cuales retornan la información detallada y actualizada sobre el estado de cada token asociado a dicha tarjeta. La información es almacenada para su posterior procesamiento.

```
2025-01-13T09:25:26.650-05:00  INFO 1 --- [  XNIO-1 task-2] c.c.b.g.s.AbstractGetCardInfoStateToken  : Inicia el proceso
2025-01-13T09:25:26.657-05:00  INFO 1 --- [  XNIO-1 task-2] c.c.b.g.s.GetCardInfoDeletedSuspended    : Eliminadas-Suspendidas de APPLE_PAY: 5 tarjetas
2025-01-13T09:25:28.049-05:00  INFO 1 --- [         task-2] c.c.b.g.s.AbstractGetCardInfoStateToken  : Total tokens respondidos por Thales 5
2025-01-13T09:25:28.351-05:00  INFO 1 --- [         task-2] c.c.b.g.s.AbstractGetCardInfoStateToken  : 4 de 5 procesados
2025-01-13T09:25:28.393-05:00  INFO 1 --- [         task-2] c.c.b.g.s.AbstractGetCardInfoStateToken  : Finaliza el proceso
```

Figura 6 Proceso en ejecución

Para finalizar, el sistema compara la información almacenada que fue recibida desde las franquicias con los valores almacenados en la base de datos. Procede a buscar en la base de datos los tokens que correspondan al "virtual_card_id" devuelto por las franquicias y en caso de detectar diferencias en el estado del token, se procede a actualizar tanto el estado en la base de datos como la fecha de la última modificación (Columna "date_update"), asignándole la marca temporal correspondiente al momento de la actualización. Cuando este flujo finaliza los tokens que tuvieron cambios en las franquicias y la información no llegó en su momento a la base de datos quedan con su estado actual, tal y como se observa en la Figura 7, lo que permite la correcta homologación de un aproximado de 200 a 500 tokens diarios entre franquicias y banco.

	pk_id_visa_request	issuer_card_ref_id	virtual_card_id	state_token	date_request	date_update
1	2.513.248	issuer_gtk_del_sus_apple_1	virtual_card_gtk_del_sus_apple_1	ACTIVATE	2024-12-13 09:15:13.640	[NULL]
2	2.513.251	issuer_gtk_del_sus_apple_4	virtual_card_gtk_del_sus_apple_4	SUSPEND	2024-12-13 09:15:13.640	[NULL]
3	2.513.249	issuer_gtk_del_sus_apple_2	virtual_card_gtk_del_sus_apple_2	SUSPEND	2024-12-13 09:15:13.640	2025-01-13 09:25:28.059
4	2.513.250	issuer_gtk_del_sus_apple_3	virtual_card_gtk_del_sus_apple_3	DELETE	2024-12-13 09:15:13.640	2025-01-13 09:25:28.318
5	2.513.252	issuer_gtk_del_sus_apple_5	virtual_card_gtk_del_sus_apple_5	DELETE	2024-12-13 09:15:13.640	2025-01-13 09:25:28.355

Figura 7 Datos al finalizar la ejecución del proceso de actualización

6. Análisis

La adopción de los patrones de diseño Template Method y Strategy permitió abordar dos aspectos esenciales del sistema: unificación y flexibilidad. Por un lado, el Template Method facilitó la integración de procesos que, si bien compartían comportamientos similares, requerían funcionamientos puntuales según cada necesidad. Este patrón ofreció un esqueleto común para los algoritmos y delegó la implementación de los detalles específicos en las clases hijas, con lo cual se evitó la duplicación de código y se mantuvo la coherencia en el comportamiento general del sistema. Al mismo tiempo, el uso de Strategy aportó una elevada flexibilidad al posibilitar el intercambio dinámico de distintos algoritmos durante la ejecución en tiempo real, sin modificar la lógica central. Lo cual no solo simplifica el mantenimiento, sino que también prepara la solución para evolucionar ante nuevas necesidades, transformando el sistema más escalable y adaptable a largo plazo.

Por otra parte, la estandarización y aseguramiento de la calidad se sostienen en la definición de un pipeline de despliegue y validación, donde se contemplan diversas fases para el control de la calidad del código y la detección de posibles vulnerabilidades. Herramientas como SonarCloud contribuyen a detectar defectos tempranamente y a mantener altos estándares de seguridad y mantenibilidad, mientras que las pruebas de aceptación, pruebas de rendimiento y pruebas E2E garantizan el correcto funcionamiento de la solución en escenarios reales. De esta manera, se sientan las bases para una arquitectura sólida que evoluciona con prácticas claras, ágiles y confiables haciendo uso de las mejores prácticas de la industria, asegurando así la calidad del producto de principio a fin.

7. Conclusiones

La implementación de un flujo capaz de identificar los tokens que han sido eliminados y cuya información no ha sido compartida con el banco permitió mantener una base de datos con datos actuales gracias a la actualización diaria de un aproximado de 200 a 500 tokens diarios, lo cual reforzó la consistencia de los datos que permitirá al banco tomar mejores decisiones con respecto a los costos que estos les están generando. Además, la puesta en marcha de la comunicación con franquicias facilitó la obtención de toda la información necesaria sobre dichos tokens.

La validación de este flujo a través de un pipeline de despliegue, con pruebas en entornos de desarrollo y calidad, posibilitó la detección temprana de errores y el aseguramiento de la fiabilidad del sistema. Al integrar revisiones de código mediante SonarCloud, junto con pruebas de aceptación, rendimiento y E2E, se reforzaron la mantenibilidad y la escalabilidad de la solución. Esta serie de controles y pruebas no solo garantizó el cumplimiento de los requisitos funcionales, sino también el de los no funcionales, especialmente en lo referente a la homologación de información entre el banco y las franquicias.

Para finalizar, estas acciones cumplieron con el objetivo de establecer un flujo sólido y flexible que contribuye a la comunicación y coherencia de información entre el banco y las franquicias. El resultado es una solución robusta y alineada con las metas propuestas, enfocada en una gestión eficiente de los tokens eliminados y en un proceso de despliegue confiable y seguro.

Referencias

AWS. (s/f). *¿Qué son las pruebas unitarias?* Amazon.com. Recuperado el 6 de enero de 2025, de https://aws.amazon.com/es/what-is/unit-testing/

Cabañas, K. (2023, agosto 16). *Introducción al Testing E2E*. Kranio.io. https://www.kranio.io/blog/introduccion-al-testing-e2e

Congote, E., & Hincapie, A. (2020, septiembre 7). *Pruebas Perfomance: tipos y etapas*. Pragma. https://www.pragma.com.co/academia/lecciones/pruebas-perfomance-tipos-y-etapas

Damen, A. (2020, octubre 9). ¿Qué es la Tokenización de pagos? ¿Cuáles son sus beneficios para el ecommerce? *MONEI*. https://monei.com/es/blog/what-is-tokenization-and-its-benefits-for-e-commerce/

IBM. (2024, mayo 9). *¿Qué son las pruebas de software y cómo funcionan?* Ibm.com. https://www.ibm.com/mx-es/topics/software-testing

Parra, D. (2024a, octubre 1). *Patrones de diseño - Template Method*. The power ups. https://thepowerups-learning.com/patrones-de-diseno-template-method/

Parra, D. (2024b, noviembre 6). *Patrones de diseño - Strategy*. The power ups. https://thepowerups-learning.com/patrones-de-diseno-strategy/

Schwaber, K., & Sutherland, J. (2020). *La Guía de Scrum*. https://scrumguides.org/docs/scrumguide/v2020/2020-Scrum-Guide-Spanish-Latin-South-American.pdf

Sommerville, I. (2011). *Ingeniería de software* (9ª ed.). Addison-Wesley, Pearson Educación.