FACULTAD DE INGENIERÍA

Carrera de Ingeniería de Sistemas Computacionales

“ARQUITECTURA MICRO-FRONTEND PARA LA OBSOLESCENCIA DE SOFTWARE DE UNA APLICACIÓN WEB FINANCIERA”

Trabajo de suficiencia profesional para optar el título profesional de:

Ingeniero de Sistemas Computacionales

Autor:
Romario Vargas Jacinto

Asesor:
Msc. Raúl Eduardo Huarote Zegarra

Lima - Perú

2022

DEDICATORIA

La presente tesis está dedicado a mis padres, Eusebio y Sonia, y mis hermanas, Noemí y Priscila, quienes han sido la motivación y el apoyo en los cumplimientos de diversos objetivos y metas. A mis padres que me enseñaron la responsabilidad y perseverancia.

AGRADECIMIENTO

Agradezco a mis padres por todo el apoyo y la paciencia que me dieron durante todos los años que duro la carrera y que me siguen brindando, asimismo, por la motivación del desarrollo de la presente tesis y poner culminar satisfactoriamente esta etapa.

Tabla de contenidos

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

RESUMEN EJECUTIVO

El trabajo abarcó la arquitectura micro-frontend que permite romper una arquitectura monolito de la aplicación web financiera del cliente bancario de la empresa TS NET S.A (EVOL). El presente estudio se origina a raíz del anuncio de Microsoft en donde indican que el navegador Internet Explorer 11, único navegador en donde funciona correctamente la aplicación web financiera ya no contaría con soporte técnico y tampoco sería compatible con algunas versiones de sistema operativo Windows 10. Por ello, a fin de eliminar la obsolescencia de software, el objetivo principal de la presente tesis fue implementar una arquitectura micro-frontend para la obsolescencia de software de la aplicación web financiera, por este motivo fue necesario crear una nueva aplicación web con la arquitectura micro-frontend que permite acceder desde un navegador moderno como Google Chrome y permitir la convivencia con la aplicación web monolito. Se utilizó el marco de trabajo ágil SCRUM y los frameworks Angular 14 y .NET 6 para la construcción de la aplicación web. Como resultado del presente proyecto, se cumplió con la implementación progresiva de la arquitectura micro-frontend de la aplicación web financiera a nivel nacional beneficiando a los representantes financieros del cliente bancario.

CAPÍTULO I. INTRODUCCIÓN

TS NET S.A (EVOL) es una empresa de servicios TI y consultora en el rubro de tecnologías de la información orientada a facilitar a las organizaciones en el despliegue satisfactorio de las estrategias empresariales permitiéndoles convertirse en más grandes, fuertes y ágiles. Por este motivo, EVOL propone, diseña y fortalece estrategias empresariales que agregar valor a las organizaciones. Además, en EVOL se cree en que la tecnología y su poder de conectar a las personas con las empresas. TS NET S.A se diferencia de sus competidores mediante la identidad EVOL, el cual es su identidad corporativa compuesta por tres elementos:

- Nuestra pasión: Por la excelencia, por el cambio y por el conocimiento.
- Nuestros valores: Siempre la verdad, Profesionalismo, Lo mejor para el cliente, Integridad y Confidencialidad.
- Nuestro talento: Para transformar, para educar y para operar.

Asimismo, la empresa cuenta con el siguiente ideario:

Misión

"Nosotros hacemos florecer compañías, haciéndola más rápidas, más fuentes y más grandes. Entendiendo la estrategia, innovando la visión operativa y entregando alta performance."

Visión

"Ser líderes por entendimiento".

Valores

- Siempre la verdad
- Profesionalismo
- Integridad
- Confidencialidad

TS NET S.A inicio sus operaciones el 15 de diciembre en el año de 1997, en la actualidad con 25 años de experiencia, en los cuales pese a circunstancias desafiantes siempre ha salido adelante con el espíritu de servicio y esfuerzo que lo identifica. A continuación, los hitos de la historia:

- 1997: Se crear TS NET S.A para desarrollar la práctica de Service Management Consulting.
- 1998: Desarrollo de ERP XRay para el mercado emergente.
- 1999: Desarrollo de servicios SAP en la base instalada.
- 2000: Desarrollo del modelo de Service Management bajo ITIL para la gran empresa.
- 2001: Apertura de oficina en México.
- 2002: Atención a clientes de Asia, Europa y Latinoamérica.
- 2003: Desarrollo de competencias en Oracle en aplicaciones y tecnología.
- 2004: Desarrollo implementaciones de Inteligencia de Negocios.
- 2005: Desarrollo de servicio sobre productos Microsoft.
- 2006: Implementación de proyectos de infraestructura y seguridad.
- 2007: Desarrollo de la Unidad de Negocios de Consultoría de Procesos.

- 2008: Apertura del centro de entrenamiento EVOL.

Actualmente, TS NET S.A (EVOL) cuenta con una trayectoria de 25 años de crecimiento y experiencia en donde ha logrado desarrollar proyecto en 3 continentes, más de 800 proyectos de consultoría con más de 200 clientes satisfechos y recurrentes. Todo esto gracias a las más de 5 millones de horas hombres utilizado entre los más de 400 consultores que cuenta EVOL. Además, EVOL se ha convertido socio de negocios de líderes tecnológicos como: Oracle, Red Hat, Microsoft Azure, IBM Watson, SAFe, HUAWEI, SAP y AWS.

En la actualidad, cuenta con sedes en Perú, México y Colombia. La sede central de la empresa TS NET S.A (EVOL) está ubicado en la avenida Javier Prado, número 2612 en la urbanización San Borja Norte Lima, en el distrito de San Borja, departamento Lima, país Perú. En México, la oficina se encuentra ubicada en Cantú 9, oficina 605 colonia Anzures en Ciudad de México. Finalmente, en Colombia, la oficina se encuentra en avenida CL 24 95 A 80, oficina 606 en Bogotá.

TS NET S.A (EVOL) tiene como giro Principal – 6202 – Consultoría de Informática y Gestión de Instalaciones Informáticas; además ofrece las siguientes soluciones TI:

- Cloud
    - Software as a Services (SaaS)
    - Infrastructure as a Service (IaaS)
    - Platform as a Service (PaaS)
    - Data as a Service (DaaS)

- Consulting
    - Gestión de la información
    - Soluciones ERP
    - Consultoría en Gestión
    - Soluciones de Industria
- Technology
    - Sistemas de Ingeniería
    - Fábrica de Servicios
    - Analytics
    - Big Data
    - Diseños de arquitectura tecnológica
    - Células de desarrollo
    - Migraciones a la nube
- Outsourcing
    - Especialistas por Industria
    - Procesos y aplicaciones
    - Especialistas en desarrollo
    - Especialistas en infraestructura

1.6 Organigrama

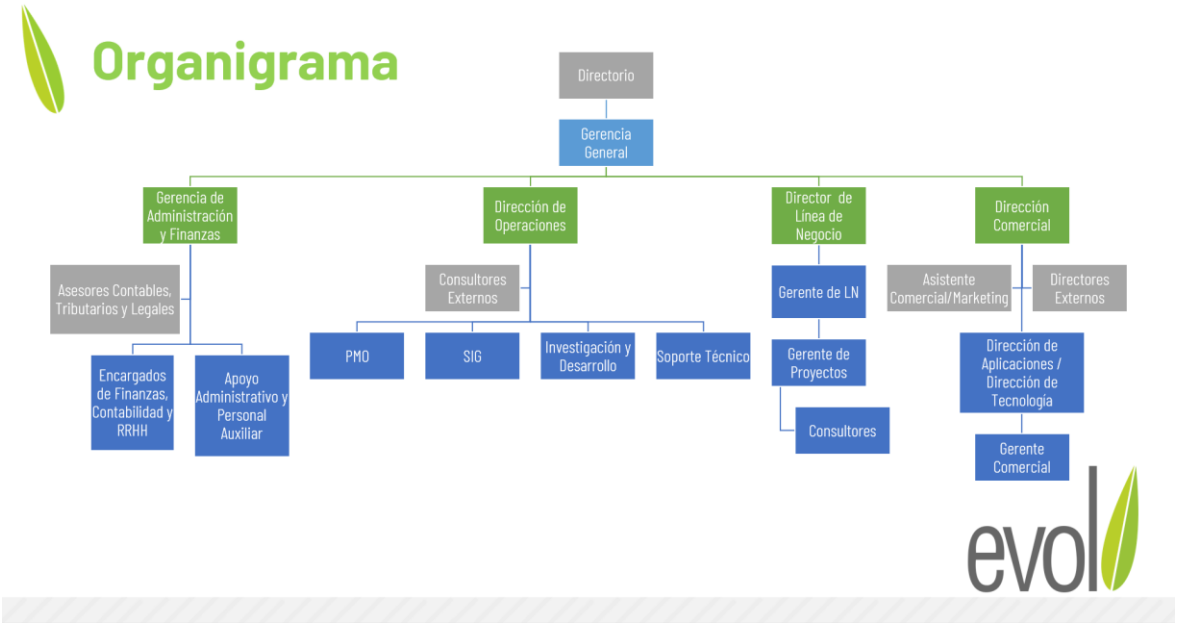


Figura 1 Organigrama de TS NET S.A (EVOL)

Fuente: TS NET S.A

Figura 2 Líneas de Negocio 2022

Fuente: TS NET S.A

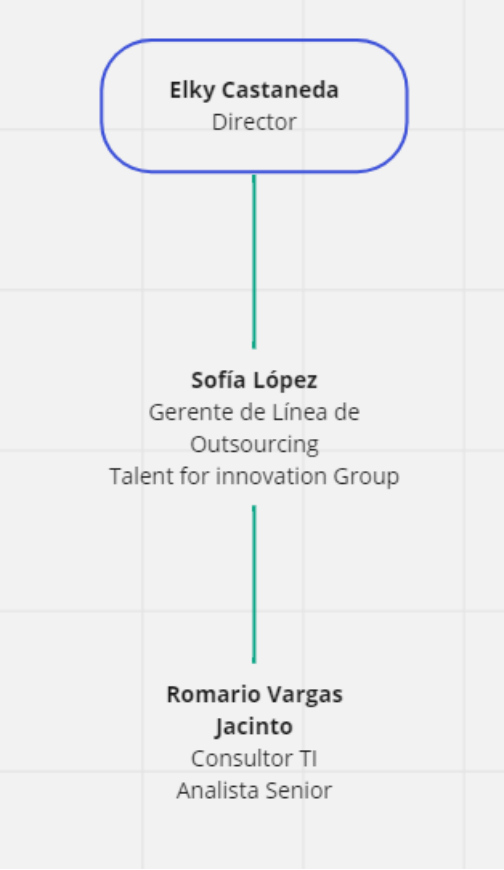


Figura 3 Organigrama de Romario Vargas

Fuente: Elaboración propia

CAPÍTULO II. MARCO TEÓRICO

A continuación, se presenta los antecedentes que motivaron la presente tesis.

2.1 Antecedentes

Nacional

El trabajo de investigación titulado *Migración de infraestructura del chatbot para mitigar el riesgo de seguridad y obsolescencia en una entidad financiera en el 2021*, de Amaro Proa (2021), publicado por la Universidad Mayor de San Marcos, en Lima, Perú, en el que concretó el objetivo del proyecto de tener el chatbot en producción y despliegue de los 14 procesos de RRHH. Además, se mitigaron los riesgos asociados a seguridad y obsolescencia. Asimismo, la migración marcó hito importante porqué permitió a la aplicación seguir evolucionando con nuevas funcionalidades sin ningún riesgo asociado. El antecedente contribuye así a reforzar a la migración de su aplicación con la finalidad de mitigar riesgos de seguridad y obsolescencia como se pretende en la presente tesis, utilizar la arquitectura micro-frontend para quitar la obsolescencia de software.

El trabajo de investigación titulado *Implementación de una arquitectura micro-frontend para optimizar el desarrollo y despliegue de la aplicación web del sistema de información universitaria de la SUNEDU*, de García (2022), publicado por la Universidad Mayor de San Marcos, en Lima, Perú, demostró que la implementación de la arquitectura micro-frontend en la aplicación web fronend del SIU no afectó las funcionalidades que ya se encontraban desarrolladas. Además, que permitió disminuir

los tiempos de compilación y despliegue de la aplicación. Asimismo, facilita un despliegue independiente de cada micro-aplicación sin que afecte a las demás aplicaciones que ya están desplegadas. El antecedente contribuye así en que con la arquitectura micro-frontend es posible renovar de forma gradual la "LA APLICACIÓN" del "EL BANCO" como se pretende en la presente tesis, el cual logró dividir a diferentes módulos

Internacional

El trabajo de titulación titulado *Análisis de la Obsolescencia del Software por el avance de la tecnología*, de Vélez Aguirre (2019), publicado por la Universidad Estatal de Milagro, en Milagro, Ecuador, demostró que los avances tecnológicos desfavorecen el uso de programas y sistemas dejándolos obsoletos. Asimismo, identificó las causas de obsolescencia de software, las cuales son: cambios no previstos en el desarrollo de un software o actualizaciones de mantenimiento. Por ende, manifestó que mantener un software obsoleto genera lentitud en su uso, incompatibilidad de hardware, problemas de seguridad y costos de mantenimiento, los cuales afecta directamente al usuario. No obstante, afirmó que, aunque no se puede evitar la obsolescencia de software se cuenta con otras opciones como la migración del software para extender la vida útil. El antecedente contribuye a la presente tesis, al determinar los riesgos de obsolescencia de software el cual se pretende eliminar mediante la arquitectura de software.

El trabajo de investigación titulado *Modelo de Arquitectura para Front-End (Basado en Micro-Frontends) Aplicado al Producto Digital de Fuerza Especializada de Vivienda del Banco de Bogotá*, de Atuesta & Nariño (2020), publicador por la

Universidad Distrital Francisco José de Caldas, en Bogotá, Colombia, demostró que era posible aplicar los principios de microservicios al frontend de una aplicación web. Asimismo, al tener una arquitectura micro-frontend es posible tener una aplicación mantenible y escalable, lo que permite que varios equipos puedan desarrollar funcionalidades específicas de la aplicación. El antecedente contribuye así a reforzar que la arquitectura micro-frontend permite construir una aplicación mantenible y escalable.

En adelantes, las bases teóricas del presente trabajo.

2.2 Bases teóricas de variable Arquitectura Micro-FrontEnd

Definición de FrontEnd

Según Pérez Ibarra, Quispe, Mullicundo y Lamas (2021) "El FrontEnd trabaja la interfaz visual, y hace que el usuario pueda interactuar con nuestro sitio o sistema. Está orientado al lenguaje de marcas y al lenguaje de programación web de ejecución en equipos clientes.". Además, los autore recalcaron que el "FrontEnd se encarga de estilizar la página de tal manera que la misma pueda presentar la información de forma agradable para el usuario." (Pérez Ibarra, Quispe, Mullicundo, & Lamas, 2021, págs. 347 - 348).

Asimismo, Recio García (2016) señalo que "La parte del cliente encargada de obtener los datos y visualizarlos se denomina front-end. Utiliza numerosas tecnologías de las que ya hemos hablado: HTML, CSS, JavaScript, JQuery, AJAX…". (pág. 19).

De acuerdo con lo expuesto, podemos precisar que el FrontEnd hace referencia a la capa de presentación o interfaz visual de una aplicación con el cual el usuario interactúa directamente. Para el caso de un aplicativo web, se utilizan tecnologías principales como HTML, CSS y JavaScript.

Tecnologías en el FrontEnd

JavaScript

Gutiérrez Gonzáles y López Goytia (2016) indicaron que JavaScript

> Es un lenguaje de programación inter-pretado, sencillo, orientado a objetos, basado en prototipos, imperativo y dinámico, que permite la ejecución de código dentro de las páginas en HTML; su ejecución por lo ge-neral es rápida. Es uno de los lenguajes de programación más utilizados para desarrollo del lado del cliente, al que también se le han hecho mejoras y adiciones de utilerías que facilitan el desarrollo del lado del servidor. (pág. 61).

Zofío Jiménez (2013) también señala que JavaScript "Es un lenguaje de programación interpretado, por lo que no es necesario compilar los programas para ejecutarlos, sino que es el propio navegador quien lo interpreta al cargar la página o al interactuar el usuario con la misma." (pág. 53).

Recio García (2016) ratificó que

> JavaScript es un lenguaje ejecutado por el navegador. Su sintaxis es similar a la del archiconocido lenguaje de programación Java -de ahí su nombre-. La idea básica de JavaScript es permitir definir acciones cuando ocurren ciertos

> eventos en el navegador: la página se ha cargado completamente, el usuario hace un clic, etc. (pág. 24).

Adicionalmente, Groner (2014) afirma que "JavaScript is a very powerful language. It is the most popular language in the world and is one of the most prominent languages on the Internet." [JavaScript es un lenguaje muy potente. Es el lenguaje más popular del mundo y es uno de los más destacados en Internet.] (pág. 26).

De lo expuesto, se concluye que JavaScript es un lenguaje principal para el desarrollo del FrontEnd de una aplicación web, pues es el único que un navegador web puede interpretar, lo que lo hace un lenguaje muy popular y destacado.

TypeScript

TypeScript es un lenguaje de programación que Microsoft desarrolló. De acuerdo con Microsoft (s.f.) "TypeScript is a strongly typed programming language that builds on JavaScript, giving you better tooling at any scale." [TypeScript es un lenguaje de programación fuertemente tipado que se basa en JavaScript, ofreciéndole mejores herramientas a cualquier escala.].

Asimismo, Chiraretta (2018) indica que "… Microsoft released TypeScript. It introduces support for classes, modules, and arrow syntax as proposed by ES6 and also other concepts that are currently not available in JavaScript, such as strong typing, interfaces, generics, and more." [… Microsoft ha lanzado TypeScript. Introduce el soporte para clases, módulos y la sintaxis de flecha propuesta por ES6 y también otros

conceptos que actualmente no están disponibles en JavaScript, como la tipificación fuerte, las interfaces, los genéricos, y más] (pág. 42).

Angular

Se puede definir (Google, s.f.) como "un framework de diseño de aplicaciones y plataforma de desarrollo para crear aplicaciones de una sola página eficientes y sofisticadas.".

En 2018, Chiaretta señala que:

> The Angular framework is currently maintained by Google together with a community of individual and corporate developers. Angular is a client-side framework built around the idea of extending HTML with new elements called web components that add additional behaviors. Components can be either HTML attributes or elements. [El framework Angular es mantenido actualmente por Google junto con una comunidad de desarrolladores individuales y corporativos. Angular es un framework del lado del cliente construido en torno a la idea de extender el HTML con nuevos elementos llamados componentes web que añaden comportamientos adicionales. Los componentes pueden ser atributos o elementos de HTML] (págs. 43 - 44).

Además, de acuerdo con la página oficial de Angular, indica que es una moderna plataforma para desarrolladores web que permite desarrollar para todas las plataformas utilizando el TypeScript.

Angular Material

Angular Material son componentes de Material Design para angular que permiten utilizar componentes internacionalidades garantizando el rendimiento y la fiabilidad proporcionado herramientas que facilitan a los desarrolladores construir componentes propios y personalizados. (Angular Components Team, s.f.).

Definición de BackEnd

"Se denomina BackEnd a la capa de acceso a los datos de un software que no es accesible para el usuario final. Además, esta capa contiene toda la lógica de la aplicación que maneja los datos." (Pérez Ibarra, Quispe, Mullicundo, & Lamas, 2021, pág. 348).

Por ello, Pérez Ibarra, Quispe, Mallicundo y Lamas (2021) recalco que "El Backend se encarga de la manipulación de los datos, un BackEnd no sirve de mucho si no existe un FrontEnd de por medio, el desarrollador BackEnd debe de conocer de bases de datos, frameworks y aspectos de seguridad." (pág. 347).

Además, a diferencia del FrontEnd que utiliza tecnologías orientado a la interacción del usuario, "Por otro lado, el servidor -denominado back-end- utiliza otras tecnologías más orientadas al almacenamiento y procesado de datos." (Recio García, 2016, pág. 19).

En vista de lo anterior, podemos concluir que el BackEnd es parte de una aplicación que nos es visible a los usuarios y el cual está asignado al almacenamiento y

manipulación de datos. Por tal motivo, el lenguaje de programación que utiliza es distinta a del FrontEnd de la aplicación.

Tecnologías en el BackEnd

C#

De acuerdo con Gutiérrez Gonzáles y López Goytia (2016), C# "Es un lenguaje de programación orientado a objetos creado por Microsoft, que se puede entender como una versión mejorada de C y C++ y está integrado a la plataforma .NET." (pág. 67).

En la página de Microsoft (Wagner, y otros, 2022) indica que:

> C# (pronunciado "si sharp" en inglés) es un lenguaje de programación moderno, basado en objetos y con seguridad de tipos. C# permite a los desarrolladores crear muchos tipos de aplicaciones seguras y sólidas que se ejecutan en .NET. C# tiene sus raíces en la familia de lenguajes C, y a los programadores de C, C++, Java y JavaScript les resultará familiar inmediatamente.

.NET

"NET is a free, cross-platform, open source developer platform for building many different types of applications.". [.NET es una plataforma para desarrolladores gratuita, multiplataforma y de código abierto que permite crear muchos tipos diferentes de aplicaciones.] (Microsoft, s.f.).

En ese mismo sentido, Gutiérrez Gonzáles y López Goytia (2016) indica que .NET:

> Es una plataforma desarrollada por Microsoft que agrupa una serie de herramientas de desarrollo y componentes. Facilita la creación de servicios web y aplicaciones Windows, entre otras. Es visto como una infraestructura digital que está integrada por un CLR (Common Language Runtime), el .NET Framework Base Clases, así como ADO.NET (acceso a bases de datos), ASP.NET (genera páginas activas) y WinForms (para cons-truir aplicaciones propias para Windows). (pág. 66).

Definición de Microservicios

Roldán Martínez, Valderas Aranda y Torres Bosch (2018) indicaron que:

> "La arquitectura basada en microservicios tiene su fundamente en la división en módulos independientes denominados microservicios, que se construyen alrededor de funcionalidades de negocio muy concretas (autenticación, directorio, correo, CRUD de ítems, etc.) y que se comunican entre sí mediante mecanismos de interacción relativamente sencillos, como puedan ser llamadas RESTfull a un API o algún tipo de RPC (Remote Procedure Communications)." (pág. 21).

Por otro lado, Yang, Lui y Su (2019) definieron:

> Microservices are a variant of the service-oriented architecture architectural style that builds applications as a collection of loosely coupled services. It combines complex large applications in a modular way based on small functional blocks that communicate through a collection of language

> independent APIs. [Los microservicios son una variante del estilo arquitectónico de la arquitectura orientada a servicios que construye aplicaciones como una colección de servicios débilmente acoplados. Combina aplicaciones complejas de gran tamaño de forma modular basadas en pequeños bloques funcionales que se comunican a través de una colección de APIs independientes del lenguaje.] (pág. 2).

Asimismo, Krishnamurthy (2019) señalo que:

> The idea is to have small autonomous services to work together to build a large complex application, it mainly focusses on building individual sub-domains and small services making them easier to maintain, and promotes independently deployable pieces, thus ensuring the internal changes on one service do not affect or require the redeployment of other services. [La idea es tener pequeños servicios autónomos para trabajar juntos para construir una gran aplicación compleja, se centra en la construcción de subdominios individuales y pequeños servicios haciéndolos más fáciles de mantener, y promueve piezas desplegables de forma independiente, asegurando así que los cambios internos en un servicio no afecten o requieran el redespliegue de otros servicios.] (pág. 105).

Geers (s.f.) destacó que:

> La idea detrás de Micro Frontends es pensar en un sitio web o aplicación web como una composición de características que son propiedad de equipos independientes. Cada equipo tiene un área de negocio definida o misión de la que se preocupa y se especializa. Un equipo es cross functional y desarrolla

sus características end-to-end, desde la base de datos hasta la interfaz de usuario.

Definición de Micro-FrontEnd

Peltonen, Mezzalira y Taibi (2021) indicaron que "Micro-Frontends [1][2][3][4] were introduced in 2016 [1] to enable the decomposition of the front-end into individual and semi-independent frontends, separating the business logic from the frontend, and creating independent services that interact together [5]." [Los Micro-Frontends fueron introducidos en 2016 para permitir la descomposición del front-end en front-ends individuales y semi-independientes, separando la lógica de negocio del front-end, y creando servicios independientes que interactúan juntos.] (pág. 2)

Por ende, Yang, Liu y Su (2019) determinaron que:

> Micro frontends extends the concepts of microservices to the frontend world. It applies the concept of microservices to the browser side, transforming web applications from a single application to an application that combines multiple small front-end applications. Each front-end application can run, developed, and deployed independently. [Micro frontends extiende los conceptos de microservicios al mundo del frontend. Aplica el concepto de microservicios al lado del navegador, transformando las aplicaciones web de una sola aplicación a una aplicación que combina múltiples aplicaciones frontales pequeñas. Cada aplicación front-end puede ejecutarse, desarrollarse y desplegarse de forma independiente".] (pág. 2)

De igual forma, Deloitte (s.f.) resalto que:

> Esta arquitectura frontend, la podemos integrar con diferentes arquitecturas del Backend, para optimizar aún más sus resultados. Un ejemplo claro que nos podemos encontrar en algunas empresas, es el uso conjunto del patrón de Backend For Frontend con Microservicios y el Microfrontend.

Beneficios de Micro-FrontEnd

Yang, Liu y Siu (2019) preciso que:

> The idea behind micro frontends is to treat a web application as a combination of features owned by different teams. Each team has an independent business or function that they focus on. A team is cross-functional and develops its features end-to-end, from backend to frontend [4]. [La idea detrás de los micro frontales es tratar una aplicación web como una combinación de características propiedad de diferentes equipos. Cada equipo tiene un negocio o función independiente en el que se centra. Un equipo es multifuncional y desarrolla sus características de principio a fin, desde el backend hasta el frontend] (pág. 2)

Por otro lado, Geers (2020) indicó que "Autonomy is one of the critical benefits of microservices and also of micro frontends. It comes in handy when teams decide to make more significant changes …" [La autonomía es uno de los beneficios críticos de los micro-servicios y también de los micro-frontends. Viene muy bien cuando los equipos deciden hacer cambios más significativos...] (pág. 16).

Por otra parte, Peltonen, Mezzalira y Taibi (2021) destacan:

> Micro-Frontends extends the concepts of Microservices to the frontend side of the application. It transforms monolithic web applications from a single code based application architecture to an application that combines multiple small frontend applications into one whole. Each of these independent applications can run, and be developed and deployed independently. The capability of independent development and deployment allows development teams to build isolated and loosely coupled services. [Micro-Frontends extiende los conceptos de Microservicios al lado del frontend de la aplicación. Transforma las aplicaciones web monolíticas de una arquitectura de aplicación basada en un único código a una aplicación que combina múltiples aplicaciones frontales pequeñas en un todo. Cada una de estas aplicaciones independientes puede ejecutarse, desarrollarse y desplegarse de forma independiente. La capacidad de desarrollo y despliegue independiente permite a los equipos de desarrollo construir servicios aislados y poco acoplados] (pág. 8).

Por último, Adservio team (2022) manifestó que “Micro frontends allow End-to-End Teams of frontend, backend, and database professionals working toward a common goal, breaking down the backend and frontend monolith paradigm.” [Los micro-frotends permiten que los equipos de profesionales de frontend, backend y bases de datos trabajen hacia un objetivo común, rompiendo el paradigma del monolito de backend y frontend].

2.3 Bases teóricas de variable Obsolescencia de Software

Definición de Obsolescencia

Según Latouche (2014), “La palabra <<obsolescencia>> aparece con el cambio al siglo xx cuando unos modernos aparatos electrodomésticos empiezan a reemplazar las antiguas estufas y chimeneas.” (pág. 26).

Además, “The English word obsolescence is derived from the Latin term obsolescere, which means “to go out of use or fashion.” The associated adjective obsolescent is derived from the Latin term obsoletus, meaning “worn out”.” [La palabra inglesa obsolescence deriva del término latino obsolescere, que significa "pasar de moda o de uso". El adjetivo asociado obsolescente deriva del término latino obsoletus, que significa "gastado".] (Bartels, Ermel, Pecht, & Sandborn, 2012, págs. 1 - 2).

Adicionalmente, Vega Antonio (2012) precisó que “La obsolescencia es un término que se refiere a la vida útil, o valor de uso, de un artefacto o servicio en función del tiempo, y en el contexto económico se asocia con la depreciación.” (pág. 56).

Asimismo, Dueñas Ramírez y Villegas López (2020), aclaró que:

> En general, la obsolescencia es un concepto que hace referencia a algo que, dadas sus características, ya queda fuera de uso o de práctica. Generalmente se asocia a bienes materiales, los cuales tienen una funcionalidad y cuando esta se pierde total o parcialmente, dicho bien pierde su objetivo de ser. (pág. 3).

Definición de Software

Rajagopal, Erkoyuncu y Roy (2014) señalaron que "Software is defined as programs, procedures, rules, data and docu mentation associated with programmable aspects of system hardware and infrastructure." [El software se define como programas, procedimientos, reglas, datos y documentación asociados a los aspectos programables del hardware y la infraestructura del sistema] (pág. 76).

Definición de Obsolescencia de Software

"Software obsolescence happens when the original developer and authorised third party cease to provide support with regular updates, upgrades, and fixes or due to changes in the target environment, systems, and hardware, which makes software unusable." [La obsolescencia del software se produce cuando el desarrollador original y la tercera parte autorizada dejan de prestar apoyo con actualizaciones, mejoras y correcciones periódicas o debido a cambios en el entorno, los sistemas y el hardware de destino, lo que hace que el software sea inutilizable.] (Rajagopal, Erkoyuncu, & Roy, 2014, pág. 76).

2.4 Definición de términos básicos

Jira Software

Este software "se ha convertido en una potente herramienta de gestión de trabajo para todo tipo de casos de uso, desde la gestión de requisitos y casos de prueba hasta el desarrollo de software ágil." (Atlassian, s.f.).

Bitbucket

"Bitbucket Cloud es una herramienta de alojamiento de código y colaboración basada en Git diseñada para equipos." (Atlassian, s.f.). Además, Bitbucket tiene integración con Jira Software, de esta manera, se puede asociar las tareas del tablero scrum del equipo de desarrollo con ramas o pull request de un repositorio de código fuente (Atlassian, s.f.).

InVision

Este software es útil pues en forma de un lienzo digital permite diseñar prototipos de alta fidelidad de forma colaborativo, permitiendo que diferentes usuarios puedan crear los diseños y evaluarlos. (Invision, s.f.).

PlantUML

Este software "es un proyecto Open Source (código abierto) que permite escribir rápidamente" (PlantUML, s.f.) diferentes tipos de diagramas, como de secuencia, casos de uso, clases u objectos. El resultado de los diagramas puede ser exportados como archivos en formato PNG, SVG o LaTeX.

Visual Code

Microsoft (s.f.) indica que:

> Visual Studio Code is a lightweight but powerful source code editor which runs on your desktop and is available for Windows, macOS and Linux. It comes with built-in support for JavaScript, TypeScript and Node.js and has a rich ecosystem of extensions for other languages and runtimes (such as C++, C#, Java, Python, PHP, Go, .NET). [Visual Studio Code es un editor de código fuente ligero pero potente que se ejecuta en el escritorio y está disponible para Windows, macOS y Linux. Viene con soporte incorporado para JavaScript, TypeScript y Node.js y tiene un rico ecosistema de extensiones para otros lenguajes y tiempos de ejecución (como C++, C#, Java, Python, PHP, Go, .NET).]

Entity Framework

"Entity Framework Core is a modern object-database mapper for .NET. It supports LINQ queries, change tracking, updates, and schema migrations. EF Core works with many databases, including SQL Database (on-premises and Azure), SQLite, MySQL, PostgreSQL, and Azure Cosmos DB." [Entity Framework Core es un moderno mapeador de bases de datos de objetos para .NET. Admite consultas LINQ, seguimiento de cambios, actualizaciones y migraciones de esquemas. EF Core funciona con muchas bases de datos, como SQL Database (local y Azure), SQLite, MySQL, PostgreSQL y Azure Cosmos DB.] (Microsoft, s.f.).

Visual Studio 2022

Microsoft (s.f.) indica que:

> Visual Studio 2022 es el mejor Visual Studio. Nuestro primer IDE de 64 bits facilita el trabajo con proyectos aún más grandes y cargas de trabajo más complejas. Las cosas que hace todos los días, como escribir código y cambiar de rama, se sienten más fluidas y responden mejor.

Postman

"Postman is an API platform for building and using APIs. Postman simplifies each step of the API lifecycle and streamlines collaboration so you can create better APIs—faster." [Postman es una plataforma de API para crear y utilizar APIs. Postman simplifica cada paso del ciclo de vida de la API y agiliza la colaboración para que puedas crear mejores APIs, más rápido.] (Postman, s.f.).

SQL Server 2016

> Microsoft SQL Server is a relational database management system (RDBMS) that supports a wide variety of transaction processing, business intelligence and analytics applications in corporate IT environments. Microsoft SQL Server is one of the three market-leading database technologies, along with Oracle Database and IBM's DB2. [Microsoft SQL Server es un sistema de gestión de bases de datos relacionales (RDBMS) que admite una amplia variedad de aplicaciones de procesamiento de transacciones, inteligencia empresarial y análisis en entornos informáticos corporativos. Microsoft SQL

> Server es una de las tres tecnologías de bases de datos líderes del mercado, junto con Oracle Database y DB2 de IBM.] (Hughes, 2019)

Además, en SQL Server es posible implementar almacenamiento en Cache. “The Distributed SQL Server Cache implementation (AddDistributedSqlServerCache) allows the distributed cache to use a SQL Server database as its backing store.” [La implementación de la caché distribuida de SQL Server permite que la caché distribuida utilice una base de datos de SQL Server como almacén de respaldo.] (Anderson, y otros, 2022).

SOLID

“The SOLID Principles are five principles of Object-Oriented class design. They are a set of rules and best practices to follow while designing a class structure.” [Los Principios SOLID son cinco principios de diseño de clases orientadas a objetos. Son un conjunto de reglas y mejores prácticas a seguir mientras se diseña una estructura de clases.] (Erinç, 2020).

> Algunos principios de desarrollo de software fantásticos se han reunido bajo el acrónimo SOLID: Single responsibility (responsabilidad única), Open for extension and closed for modification (abierto a extensiones y cerrado a modificaciones), Liskov substitution (sustitución de Liskov), Interface segregation (segregación de interfaces) y Dependency injection (inyección de dependencias). (King, 2015).

Clean Code

"Clean coding is not a skill that can be acquired overnight. It is a habit that needs to be developed by keeping these principles in mind and applying them whenever you write code." [La codificación limpia no es una habilidad que pueda adquirirse de la noche a la mañana. Es un hábito que hay que desarrollar teniendo en cuenta estos principios y aplicándolos siempre que se escriba código.] (Erinç, 2020).

Microsoft Azure

Azure es un conjunto de servicios en la nube en expansión constante que ayudan a la organización a cumplir los desafíos empresariales actuales y futuros. Azure le ofrece la libertad de compilar, administrar e implementar aplicaciones en una red global masiva mediante sus herramientas y plataformas favoritas. (Microsoft, s.f.)

IIS

Internet Information Services (IIS) is software that supports the operation of web servers. IIS includes a set of web server feature modules, also called extensions. The extensions are mutually independent and can be installed or uninstalled individually so that website administrators can customize their sites as needed. Each extension supports a functional area such as HTTP, security, diagnostics, compression, caching, and web content. [Internet Information Services (IIS) es un software que soporta el funcionamiento de los servidores web. IIS incluye un conjunto de módulos de características del

servidor web, también llamados extensiones. Las extensiones son independientes entre sí y pueden instalarse o desinstalarse individualmente para que los administradores de los sitios web puedan personalizarlos según sus necesidades. Cada extensión da soporte a un área funcional como HTTP, seguridad, diagnóstico, compresión, caché y contenido web.] (Microsoft, s.f.)

JFrog Artifactory

"JFrog Artifactory is the single solution for housing and managing all the artifacts, binaries, packages, files, containers, and components for use throughout your software supply chain." [JFrog Artifactory es la solución única para albergar y gestionar todos los artefactos, binarios, paquetes, archivos, contenedores y componentes para su uso a lo largo de su cadena de suministro de software.] (JFrog Ltd, 2016)

Node

"Ideado como un entorno de ejecución de JavaScript orientado a eventos asíncronos, Node.js está diseñado para crear aplicaciones network escalables." (Node.js, s.f.)

Internet Explorer 11

Internet Explorer (or IE) is a free graphical browser maintained by Microsoft for legacy enterprise uses. Microsoft Edge is currently the default Windows browser. Microsoft first bundled IE with Windows in 1995 as part of the package called "Microsoft Plus!". By around 2002, Internet Explorer had become the most used browser in the world, but has since lost ground to Chrome, Firefox, Edge, and Safari. [Internet Explorer (o IE) es un navegador gráfico gratuito mantenido por Microsoft para usos empresariales heredados.

> Microsoft Edge es actualmente el navegador por defecto de Windows. Microsoft incluyó por primera vez IE con Windows en 1995 como parte del paquete llamado "Microsoft Plus!". Alrededor de 2002, Internet Explorer se convirtió en el navegador más utilizado del mundo, pero desde entonces ha perdido terreno frente a Chrome, Firefox, Edge y Safari.] (Mozilla Corporation's, 2022)

Google Chrome

Es el navegador web creado por Google y es posible ejecutarse en diferentes sistemas operativos. En el caso de Windows, Google Chrome está disponible para las versiones 11/10/8.1/8/7 de 64 bits. (Google, s.f.)

CAPÍTULO III. DESCRIPCIÓN DE LA EXPERIENCIA

3.1 Acuerdo de Confidencialidad

La presente tesis bajo la modalidad de trabajo de suficiencia laboral se desarrolló en un cliente bancario de la empresa TS NET S.A. (EVOL).

Debido a contratos y cláusulas de confidencialidad, aquí en adelante se le denominará al cliente como “EL BANCO” y a la aplicación web involucrada se le denominará como “LA APLICACIÖN”.

3.2 Situación Problemática

Como consecuencia de la transformación digital, muchas organizaciones han tenido la necesidad de migrar las herramientas manuales que utilizaban para manipular y almacenar su información y conocimiento a herramientas digitales. En vista de esto, ha sido necesario la construcción software, “Conjunto de programas, instrucciones y reglas informáticas para ejecutar ciertas tareas en una computadora.” (Real Academia Española, 2022).

En vista de esto, a nivel mundial muchas organizaciones han invertido una gran cantidad de dinero en el alquiler, construcción o compra de software que les permita ser eficientes en sus tareas. Por ejemplo, “Worldwide IT spending is projected to total $4.4 trillion in 2022, an increase of 4% from 2021, according to the latest forecast by Gartner, Inc” [se prevé que el gasto mundial en TI ascienda a 4,4 billones de dólares en 2022, lo que supone un aumento del 4% respecto a 2021, según las últimas previsiones de Gartner, Inc.] (Gartner, 2022).

No obstante, aunque las aplicaciones informáticas han sido de gran ayuda para las labores de las personas, “las aplicaciones también envejecen y necesitan un cuidado continuo para seguir siendo útiles: hay que corregir errores, mejorar su seguridad, adaptarlas a nuevos dispositivos, añadir funcionalidades y hasta vigilar que cumplan con las nuevas leyes”. (Clarisó & Cabot, 2018).

Además, es interesante saber que “cuando una aplicación se crea, el desarrollador suele planificar un periodo limitado y corto de mantenimiento activo. Al final de este periodo se produce el end-of-life, es decir, el fin de la vida útil del producto.” (Clarisó & Cabot, 2018).

Del mismo modo, Rajagopal, Erkoyuncu y Roy (2014) consideraron que en caso de que el desarrollador original de un software deja de brindar apoyo en las actualizaciones, mejoras y corrección de la aplicación entonces se convierte en software inútil y por ende genera la obsolescencia de software.

Por tanto, el hecho de invertir una gran cantidad de dinero en un software y luego quede este en obsolescencia, sin duda es un grave problema para las organizaciones.

Un caso de obsolescencia de software que ha afectado a muchas organizaciones es relacionado con el navegador Internet Explorer de la empresa tecnológica Microsoft. El anuncio de Microsoft (2022) indicó que “A partir del 15 de junio de 2022, la aplicación de escritorio Internet Explorer 11 ya no será compatible con ciertas versiones de Windows 10*.”; dando así por finalizado su soporte técnico y

recomendaba cambiarse al navegador Microsoft Edge, el cual tendría soporte técnico cuando menos hasta 2029.

Este fin de soporte del navegador generó un problema en el cliente bancario de la empresa TS NET S.A. (EVOL) al cual llamaremos "EL BANCO". "EL BANCO" cuenta con una aplicación web financiera llamada "LA APLICACIÓN" que es utilizado por los representantes financieros que les permite ejecutar transacciones financieras como depósitos, retiros, pago de servicios, compra venta de moneda extranjera, etc; a petición de los clientes naturales o jurídicos. Actualmente, 2399 representantes financieros hacen uso de "LA APLICACIÓN", distribuidos en 195 tiendas financieras.

"LA APLICACIÓN" fue implementando en "EL BANCO" en el año 2001 como una aplicación web cliente servidor con 85 módulos transaccionales y una arquitectura monolito, como se aprecia en el anexo N^0 2. Actualmente, a "LA APLICACIÓN" solo es posible acceder mediante el navegador Internet Explorer versión 11 bajo la configuración de vista de compatibilidad, modo de documento Internet Explorer versión 5. Asimismo, para su correcto funcionamiento es necesario la instalación de bibliotecas de vínculos dinámicos, archivos dll's, y Java JRE versión 7u51 pues "LA APLICACIÓN" hace uso de ActiveXObject y Applet's.

Con respecto a los Applets, Oracle (s.f.) en la documentación de Java sobre Applet indica que "The APIs in this package are all deprecated. Alternative technologies such as Java Web Start or installable applications should be used instead." [Todas las API de este paquete están obsoletas. En su lugar deben utilizarse tecnologías alternativas

como Java Web Start o aplicaciones instalables.]. De igual modo, en cuanto a los ActiveX, Microsoft (s.f.) respondió que los ActiveX no es admitido por el navegador Microsoft Edge, navegador moderno baso en Chromium. En los navegadores modernos, basados en Chromium, como Microsoft Edge (Ver anexo N^0 3), Google Chrome (Ver anexo N^0 4) y Firefox (Ver anexo N^0 5) no es posible ejecutar "LA APLICACIÓN".

De acuerdo con lo expuesto, aunque "EL BANCO" invirtió dinero y esfuerzo en la implementación de la aplicación web "LA APLICACIÓN" para efectuar sus operaciones financieras, desafortunadamente ha quedo en obsolescencia de software. "LA APLICACIÓN" es obsoleto debido al fin de soporte del navegador Internet Explorer de parte de Microsoft y que no es posible acceder mediante navegadores modernos, afectando así a 195 tiendas financieras y 2399 representantes financieros.

Por ello, existe la necesidad renovar la tecnología y la arquitectura de "LA APLICACIÓN" a fin de mantenerse en vigencia y salir de la obsolescencia de software. Una arquitectura que permitirá la renovación de "LA APLICACIÓN" es la arquitectura micro-frontend pues permite el desarrollo de la aplicación por módulos.

3.3 Formulación del Problema

¿De qué manera la arquitectura micro-frontend evita la obsolescencia de software de la aplicación web financiera "LA APLICACIÓN"?

3.4 Objetivos

Objetivo General

Implementar una arquitectura micro-frontend para la obsolescencia de software de la aplicación web financiera.

Objetivos Específicos

- Realizar el cambio progresivo equivalente mediante la arquitectura micro-frontend para reducir la obsolescencia de software de la aplicación web financiera.
- Reducir el riesgo tecnológico mediante la arquitectura micro-frontend para la obsolescencia de software de la aplicación web financiera.
- Refactorización de la aplicación monolito mediante la arquitectura micro-frontend para la obsolescencia de software de la aplicación web financiera.

3.5 Justificaciones

El presente trabajo de suficiencia profesional tiene por objetivo dar a conocer la importancia de la arquitectura micro-frontend y su implementación en la aplicación web bancaria “LA APLICACIÓN” del cliente “EL BANCO” a fin de evitar la obsolescencia de software. Al utilizar la arquitectura micro-frontend permitirá que “LA APLICACIÓN” sea una aplicación mantenible, escalable en el tiempo y evitar riegos de seguridad. Además, esta arquitectura ofrece la posibilidad de ir renovación “LA APLICACIÓN” en pequeños módulos o aplicaciones.

El proyecto permitirá tener un conocimiento acerca del problema general actual de "EL BANCO", así como la formulación, el desarrollo e implementación de la solución. Asimismo, el producto tecnológico final beneficiará a los representantes financieros del "EL CLIENTE" a contar con una herramienta actualizada para realizar las operaciones financieras en navegadores modernos.

Finalmente, el presente proyecto busca demostrar la capacidad, las habilidades y las experiencias como bachiller de la carrera de ingeniería de Sistema Computacionales de analizar, desarrollar e implementar proyectos TI a fin de demostrar que se está apto para obtener el título de Ingeniero de Sistemas Computacionales.

detalladamente cuándo y cómo fue el proceso de ingreso a la empresa, mencionar

3.6 Alcance

Alcance Funcional

La aplicación financiera "LA APLICACIÓN" consta de 83 módulos transaccionales. No obstante, el alcance de la experiencia profesional es el producto viable mínimo el cual es solo un módulo de transacción financiera.

Asimismo, un requisito funcional es mantener las funcionalidades actuales de "LA APLICACIÓN".

3.7 Limitaciones

El presente proyecto no abarcará la corrección de errores existentes en programas externos con los que la aplicación web "LA APLICACIÓN" tiene comunicación

como: programas de Mainframe HOST o servicios web de IBM Integration BUS. Además, no se cambiará el flujo de negocios de las transacciones financieras.

Asimismo, no se describirá de implementación de micro-frontends de otras transacciones diferentes al Producto Mínimo Viable MVP de la aplicación "LA APLICACIÓN".

3.7 Etapas de desarrollo del Proyecto

Análisis de la situación Actual

En el año 2021, se realizó los análisis de la arquitectura y funcionamiento tanto de la aplicación web "LA APLICACIÓN" y un módulo transaccional Transacción 011 – Ministerio de Relaciones Exteriores RREEE con el fin de identificar los componentes y comportamiento. Además, este análisis fue base para diseñar, implementar o adaptar los componentes actuales con la arquitectura micro-frontend.

"LA APLICACIÓN"

La aplicación "LA APLICACIÓN" del cliente "EL BANCO" fue implementado como un sistema web cliente servidor. "LA APLICACIÓN" está compuesta de diversas aplicaciones alojados en un servidor de Windows Server 2016; los cuales se comunican mediante colas de mensajes de Windows MSMQ versión 10. Asimismo, "LA APLICACIÓN" cuenta con base de datos relacionales alojadas en instancia de SQL Server 2016 y datos maestros están en archivos XML.

A continuación, las aplicaciones principales involucradas en "LA APLICACIÓN" son:

- Aplicación web
- COM+
- Aplicación ejecutable TX_Server.exe
- Aplicación ejecutable Agente Lua
- DLL navmsgcom.dll
- DLL TX_ClientCOM.dll

Con respecto a la aplicación web de "LA APLICACIÓN" está construido bajo la tecnología de Microsoft ASP Clásico con una arquitectura monolito alojados en una sola aplicación web en el servidor Web IIS. De acuerdo con Microsoft (2017) se usa ASP "to create dynamic and interactive Web pages. An ASP page is a Hypertext Markup Language (HTML) page that contains script commands that are processed by the Web server before being sent to the client's browser." [para crear páginas web dinámicas e interactivas. Una página ASP es una página en lenguaje de marcas de hipertexto (HTML) que contiene comandos de script que son procesados por el servidor web antes de ser enviados al navegador del cliente. Esto explica el origen del término "script del lado del servidor".]. Además, se utiliza los lenguajes de scripting como VBScript y JScript para el procesamiento de datos recibidos de los formularios de la aplicación y crear instancias de las librerías de vinculo dinámico navmsgcom.dll y TX_ClientCOM.dll y COM+.

Los COM+ son utilizados para el acceso a base de datos y validación de credenciales de usuario de la aplicación web. La DLL navmsgcom.dll es utilizado para generar una trama de tipo de datos cadena con los datos del formulario. La trama generada es enviada mediante la DLL TX_ClientCOM.dll hacia la aplicación ejecutable TX_Server.exe.

La aplicación ejecutable TX_Server está desarrollado en lenguaje de programación VC++ y es el encargado de recibir las tramas enviadas por las instancias de TX_ClientCOM.dll. Las tramas recibidas son reenviadas a la aplicación ejecutable Agente LUA también desarrollada en lenguaje de programación VC++. La aplicación Agente LUA es la encarga de recibir las tramas, que fue generado en la aplicación web de "LA APLICACIÓN", y enviarlas a Mainframe-HOST para la ejecución de las transacciones mediante la aplicación HIS SNA Manager.

A la aplicación web de "LA APLICACIÓN" solo se puede acceder desde el navegador web Internet Explorer versión 11 de las computadoras de las tiendas financieras. Para ello, es necesario la instalación de DLL de controles ActiveX y Java JRE versión 7u51.

> Los controles ActiveX son pequeñas aplicaciones que permiten a los sitios web proporcionar contenido, como vídeos y juegos. También te permiten interactuar con contenido como barras de herramientas y tableros de cotizaciones mientras navegas por Internet. (Microsoft, s.f.).
>
> El applet es un programa Java™ diseñado para incluirse en un documento Web HTML. Podrá escribir un applet Java y luego incluirlo en una página HTML de una manera muy parecida a cómo se incluye una imagen. Al utilizar

un navegador habilitado para Java para ver una página HTML que contiene un applet, el código del applet se transfiere al sistema y la máquina virtual Java del navegador lo ejecuta. (IBM, 2021).

Se hace uso de DLL ActiveX para la comunicación con dispositivo impresora financiera y Applets para validación de los controles de los formularios de la aplicación web. Debido a estas tecnologías solo se pueden acceder desde el navegador Internet Explorer versión 11 bajo la configuración de vista de compatibilidad, modo de documento Internet Explorer versión 5.

Por otro lado, en cuanto a la infraestructura de "LA APLICACIÓN" cuenta con balanceo de carga y alta disponibilidad. El balanceo de carga está compuesto:

- 1 servidor virtual de balanceo de carga: El balanceo de carga está configurado con: sesiones persistentes, balanceo tipo Proxy y método de balanceo Least Connection.
- N servidores nodos de aplicaciones: Los servidores de aplicación tienen el sistema operativo Windows Server 2016 de 64 bits. Los servidores cuentan con los siguientes software base: servidor web IIS versión 10.0, MSMQ versión 10.0, Host Integration Server 2013 y la aplicación "LA APLICACIÓN" versión 4.1 - 64 bits.

La alta disponibilidad en "LA APLICACIÓN" involucra un grupo de disponibilidad AlwaysOn de Microsoft SQL Server 2016, compuesto de los siguientes:

- Testigo de compartición de archivos

- Clúster de conmutación por error de Windows Server (WSFC): El clúster cuenta con 2 tipos de replicas: 1 replica principal y 2 réplicas secundarias.
- Nodos de clúster WSFC: Son servidores que participan en un WSFC con sistemas operativos Windows Server 2016 con instancias de SQL Server 2016 que hospedan las bases de datos de "LA APLICACIÓN".

Módulo o Transacción de "LA APLICACIÓN"

Cada una de las GUIs de los módulos o transacciones de la página web de "LA APLICACIÓN" consta de:

- Archivo ASP

  Encargado de mostrar la interfaz gráfica de los formularios, gestionar las peticiones de datos de parte de los formularios y envío de tramas hacia la aplicación ejecutable TX_Server.exe utilizando instancias DLL TX_ClientCOM.dll
- Archivo JavaScript

  Encargado de la funcionalidad de la interfaz gráfica y validación de controles de los formularios.
- Archivo XML

  Este archivo contiene datos maestros para las listas desplegables de los formularios o maquetación de los formularios.

Cuando se selecciona una opción o se digita un número de transacción. La aplicación solicita la página ASP correspondiente a la transacción. La página ASP envía tramas de inicialización a la aplicación TX_Server.exe para obtener los datos de la transacción

que está almacenados en base de datos. Los datos iniciales de la transacción son retornados junto con el HTML al navegador para que se muestre la interfaz gráfica.

Una vez que se ingresan los datos en los formularios de la transacción y el usuario presionó el botón Aceptar, en el archivo JavaScript valida los datos ingresados y los envía a las páginas ASP. Nuevamente la página ASP recibe los datos de la transacción y lo envía hacía la aplicación TX_Server.exe. Una vez procesado la transacción la aplicación TX_Server.exe retorna el resultado hacia la página ASP y al navegador. Para finalizar correctamente la transacción envía trama de finalización a la aplicación TX_Server.exe.

Con respecto a la programación en los archivos se presentan los siguientes inconvenientes: código spaghetti, poca separación de capas y uso de tecnologías obsoletas como Applets y versión de JavaScript 1.5 soportada en Internet Explorer versión 5 – 8.

Una vez culminado el análisis de la situación actual de “LA APLICACIÓN” y de una de sus transacciones se llegó a la siguiente conclusión, la aplicación web de “LA APLICACIÓN” presenta obsolescencia de software. Sin embargo, la aplicación web podría ser renovada con una arquitectura micro-frontend. Esta nueva aplicación web renovada utilizaría tecnologías modernas y ser accedidos mediante un navegador web moderno basado en Chromium.

Por otro lado, las demás aplicaciones como TX_Server.exe, Agente LUA o la infraestructura de base de datos de “LA APLICACIÓN” no era necesario una actualización debido a que a la presente fecha no presentan inconvenientes con su funcionalidad y cuenta en soporte técnico respectivo.

Prueba de Concepto (PoC)

En el mes de Abril de 2021, se inició con una prueba de concepto para verificar la viabilidad del proyecto, esta PoC tuvo una duración de 3 semanas.

Como resultado de la etapa de análisis de la situación actual de "LA APLICACIÓN" del cliente "EL BANCO", se identificó que era necesario una nueva página web de la aplicación "LA APLICACIÓN" y se mantendría el uso la DLL TX_ClientCOM.dll para el envío y procesamiento de datos de las transacciones.

El alcance de la PoC fue la construcción de los siguientes módulos:

- Acceso a la aplicación:
- Transacción número 011 – Ministerio de Relaciones Exteriores RREE
- Cierre de sesión a la aplicación.

Para el desarrollo de la PoC se utilizó las siguientes tecnologías:

- Framework Angular

  Para el desarrollo interfaz gráficas de la aplicación web. Además, se eliminaría el uso de ActiveXObject y applets.

- Framework .NET

  Para el desarrollo de API's que procesarán los datos de las transacciones. Las APIS harían uso de la DLL TX_ClientCOM.dll para la comunicación de la aplicación TX_Server.dll.

La aplicación web y las API's se desplegaron a un servidor web IIS versión 10 en un servidor Windows Server 2016.

La conclusión de la PoC fue que era posible reescribir el código de "LA APLICACIÓN" en lenguajes de programación y framework's modernos. Asimismo, esto permitía acceder a la aplicación web "LA APLICACIÓN" mediante un navegador moderno basado en Chromium como el navegador Google Chrome.

Metodología

Una vez establecido que era viable el proyecto de renovación de la aplicación web de "LA APLICACIÓN", se procedió con la planificación de la metodología.

El cliente "EL BANCO" tiene un enfoque Agile en todos los equipos de su organización utilizando el marco de trabajo Scrum.

Podemos definir Scrum como "como un framework (marco de trabajo) para la gestión de productos, proyectos y servicios complejos que facilita un desarrollo mantenido e incremental." (Monte Galiano, 2016, pág. 15).

De acuerdo con Monte (2016), el framework Scrum tiene roles como: producto owner, scrum master y development teams [dueño del producto, experto en scrum y equipo de desarrollo]. Además, cuenta con los siguientes artefactos: producto backlog, sprint backlog, y burn down [pendientes de producto, pendientes de sprint e incremento]. Asimismo, considera las siguientes actividades: daily scrum, sprint review y sprint retrospective [reuniones diarias, revisión de sprint y retrospectiva de sprint].

Finalmente, se hace uso del tablero scrum en donde se refleja las historias de usuarios y tareas del equipo durante un sprint.

Sobre la base de lo descrito, el equipo formado para el presente proyecto utilizó el framework Scrum.

El desarrollo del proyecto involucra al equipo PHOENIX del Crew de Transformación Canales Presenciales del cliente “EL BANCO”, el equipo contó con los siguientes roles:

El equipo Phoenix conto con los siguientes roles:

- Producto Owner (PO)
- Tech Lead (TL): Encargado de la gestión y aseguramiento del cumplimiento de los requisitos definidos para el proceso del proyecto.
- Scrum Master (SM)
- Líder QA (LQA): Encargado de la definición de la estrategia y herramientas de las pruebas.
- QA Functional
- Líder BA & UI
- Business Analyst (BA): Encargado de proponer, diseñar y participar en las soluciones a nivel de negocio.
- Líder DevOps
- Software Engineer (SE): Lidera el diseño y construcción de los componentes Frontend (Microfrontend) y Backend de la aplicación. Guía los desarrollos asegurando la calidad del código basado en la excelencia técnica.

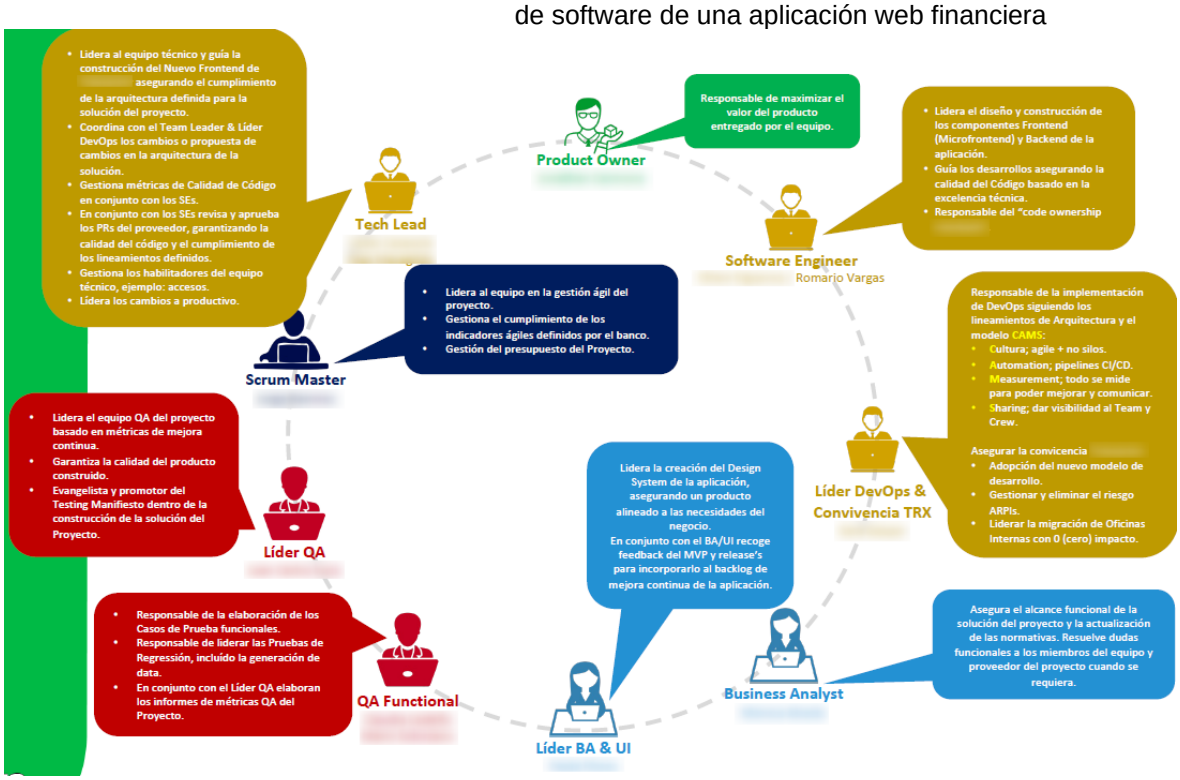


Figura 4 Roles del equipo PHOENIX
Fuente: "EL CLIENTE"

El autor de la presente tesis tuvo el rol de Software Engineer.

Además, se estableció los siguientes puntos para el presente proyecto:

- La duración de cada sprint será de dos semanas.
- Los daily scrum serán a las 9:00 am y serán liderado por cada integrante del equipo.
- Se tendrá 2 reuniones de refinamiento en cada sprint que permitirá aclarar las historias de usuarios del próximo sprint.
- El sprint review y sprint retrospective ocurrirá el último lunes del sprint por la tarde.

A continuación, las historias de usuarios que se realizaron en los diferentes sprint del alcance del presente proyecto.

Tabla 1 Historias de usuarios del proyecto

Release	Sprint	Item	Historia de Usuario
Q1	SP1	SN-1	Plan de transacciones
Q1	SP1	SN-2	Definición de arquitectura
Q1	SP2	SN-1	Creación de Design System
Q1	SP2	SN-2	Creación de template FrontEnd
Q1	SP2	SN-3	Creación de template BackEnd
Q1	SP2	SN-4	Implementación de base de datos maestros
Q1	SP4	PHOEN-5	Almacenamiento en Caché – REST API
Q1	SP4	PHOEN-59	Instalación de "LA APLICACIÓN" server v2 en servidor S50VD11
Q1	SP4	PHOEN-186	Manuales de Funcionales Generales
Q1	SP5	PHOEN-61	Habilitación de Jenkins
Q1	SP5	PHOEN-71	Solicitar asociar IP pública de Load Balancer a una DNS
Q1	SP5	PHOEN-238	Modificación de casos de Prueba R1
Q1	SP5	PHOEN-425	Setup JFrog
Q1	SP5	PHOEN-464	Revisión de normativas
Q1	SP5	SN-1	Elaboración de casos de pruebas Trx 151
Q1	SP5	SN-2	Spike Implementación de Ocelot
Q1	SP5	SN-3	Desarrollo de Api's Trx 151
Q1	SP5	SN-4	Desarrollo de micro-frontend Trx 151
Q2	SP2	PHOEN-53	Despliegue Jobs Listas Maestras UAT

Q2	SP2	PHOEN-310	Capacitaciones “LA APLICACIÓN”
Q2	SP2	PHOEN-426	Setup Azure DevOps – Parte 1
Q2	SP2	PHOEN-430	Aprovisionar server web NGNIX DEV-UAT
Q2	SP2	PHOEN-557	Pase a UAT OnPremise – Trx 151
Q2	SP2	PHOEN-560	Pase a UAT Cloud – Trx 151
Q2	SP2	PHOEN-563	Actualización / Verificación de Normativas. Trx 151
Q2	SP2	PHOEN-659	Revisión de pull request de micro-frontend Trx 151
Q2	SP3	PHOEN-562	Despliegue de Jobs Lista Maestras (Control M) a PRD
Q2	SP3	PHOEN-825	Pase a UAT OnPremise – Trx 151 – Parte 2 (Balanceador)
Q2	SP4	PHOEN-564	Certificación de la Trx 151
Q2	SP4	PHOEN-565	Despliegue de DLL Cliente PRD – Trx 151
Q2	SP4	PHOEN-969	Setup Azure DevOps – Parte 2
Q2	SP4	PHOEN-1270	Despliegue masivo DLL a nivel nacional en todas las PCs – Red de tiendas – Parte 1
Q2	SP4	RFTE-71	Ajustes del DSM para la Trx 151 – Verificación de Firmas
Q2	SP4	RFTE-214	Levantamiento de observaciones de Ethical Hacking
Q2	SP4	RFTE-221	Actualización de pruebas Automatizadas de la Trx 151 – Verificación

Q2	SP5	PHOEN-552	Aprovisionamiento Infraestructura Cloud PRD
Q2	SP5	PHOEN-1331	Despliegue masico DLL a nivel nacional en todas las PCs – Red de Tiendas – Parte 2
Q2	SP5	PHOEN-1414	Soporte pruebas de stress Trx 151 – Parte 2
Q2	SP5	RFTE-72	Configuración de Azure DevOps Pipeline Dev
Q2	SP6	PHOEN-551	Pase a PRD OnPremise – Trx 151
Q2	SP6	PHOEN-559	Pase a PRD Cloud – Trx 151
Q2	SP7	PHOEN-1772	Gestión de certificado – Malla
Q2	SP7	PHOEN-1773	Gestión de firewall Request apertura de puerto 443 - Malla

Fuente: Elaboración propia

Diseño

Arquitectura

La arquitectura elegida fue una arquitectura hibrida en donde se cuenta con una solución en la nube y de forma local.

Para la arquitectura en la nube se utilizó el Cloud Público de Microsoft Azure, en donde se tienen los siguientes componentes:

- Resource Group: Permite contener los recursos de la solución.

- Azure Blob Storage: Alberga los archivos estáticos correspondientes a los micro-frontends de las transacciones.
- Azure Container Registry
- Kubernetes Cluster
- Proxy inverso
- Azure DevOps Pipelines: Permitirá la integración y despliegue continuo de los micro-frontends.
- JFrog: Se utiliza como repositorio de artefactos universales de la aplicación.

En el caso de la arquitectura de forma local u on-premise se utilizó el Data center del cliente “EL CLIENTE”, se reutilizó la infraestructura actual, en donde se presenta lo siguiente:

- Balanceador
- Windows Server 2016
    - Servidor web IIS
    - Microsoft MSMQ
- Always On SQL Server

Esta arquitectura hace posible que la aplicación “LA APLICACIÓN” pueda ser accedida mediante las computadoras de las tiendas financieras que están en red interna de “EL CLIENTE”. El diagrama de arquitectura se aprecia en el anexo N^0 6.

Modelo de datos

La base de datos actuales de la aplicación "LA APLICACIÓN" no se modificó. Sin embarga, se creó 2 base de datos adicionales: para datos maestros que se migrarán desde archivos XMLs y otro para cache distribuida. Se utilizó SQL Server 2016 para alojar estas bases de datos. El modelo de datos se aprecia en el anexo N^0 7.

Dominio de micro-frontends

De las 83 transacciones financieras que tiene la aplicación "LA APLICACIÓN", se redujo a 43 transacciones con la aprobación del área de negocio y Product Owner.

Una vez delimitado la cantidad de transacciones, se procedió a definir los micro-frontends por dominio de negocio y los monorepo para el control de versiones de los micro-frontends, como se muestra en la figura 9.

Micro Frontend	
Dominio	**Nombre**
Bank Drafts And Travelers Checks	tmi-mf-bankdraft-traveler-check
Central Cash Handling	tmi-mf-central-cash-handling
Cheque Processing	tmi-mf-cheque-processing
Correspondent Bank Data Management	tmi-mf-correspondent-bank-data-management
Credit Card	tmi-mf-credit-card
Currency Exchange	tmi-mf-currency-exchange
Current Account	tmi-mf-current-account
Mutual Fund Administration	tmi-mf-mutual-fund-administration
Service Product	tmi-mf-service-product
Loans	tmi-mf-loans
Party Authentication	tmi-mf-party-authentication
Employee Access	tmi-mf-employee-access
Cross	tmi-mf-cross-signature tmi-mf-cross-authorization-key tmi-mf-cross-payment-methods tmi-mf-cross-pinpad tmi-mf-cross-biometria

Figura 5 Dominios de micro-frontends
Fuente: "EL CLIENTE"

Diseño UI y UX

En el caso diseño UI y UX se definió que no debería alterarse el flujo ni reglas de negocios actuales de la transacción de la aplicación "LA APLICACIÓN" y que se basaría en Angular Material para los nuevos diseños de las transacciones.

En InVisión se almacenó el diseño UX y sistemas de diseño de los componentes. En el anexo N^0 8 se muestra el diseño UX y en el anexo N^0 9 el diseño UI.

Diagrama de secuencia

Luego de la definición del diseño UI - UX se procedió a realizar diagramas de secuencias que permitió identificar los componentes backends y su comunicación, para lo cual se hizo uso de PlantUML. En el anexo N^0 10 se evidencia los diagramas de secuencias.

Construcción de Software

Micro-frontends

Se inició con la configuración del ambiente de desarrollo, para lo cual se instaló lo siguiente:

- Node

  El instalador se descargó de la página web oficinal: https://nodejs.org/es/download/.

  Una vez finalizada la descarga se procede con su instalación, una vez terminada ya contamos con node y NPM, el administrador de paquetes de node.

- Angular

Para su descarga e instalación se ejecuta el siguiente comando NPM en la terminal del sistema operativo:

npm install -g @angular/cli

- Nx Plugin

  Para su descarga e instalación se ejecuta el siguiente comando NPM en la terminal del sistema operativo:

  npm install -g nx

Una vez configurado el ambiente de desarrollo, se procedió con la creación de todos los repositorios de fuentes y su desarrollo. A continuación, los repositorios creados para el proyecto:

- tmi-mf-dsm-library

  Monorepo de toda la libraría DSM Distribuited Shared Memory o Memoria Distribuida Compartida. Contiene el tmi-mf-cross-webcomponent, libraría que agrupa a los átomos, moléculas, organismos y temas que se utilizarán en las diferentes vistas de la aplicación “LA APLICACIÓN”.

- tmi-mf-base

  Contiene los componentes de la página principal de la aplicación “LA APLICACIÓN”, como:

  - tmi-mf-base-shell
  - tmi-mf-base-footer
  - tmi-mf-base-message

- tmi-mf-base-toolbar

- tmi-mf-config-server

  Contiene los endpoint de las APIs correspondientes a los micro-frontends.

- tmi-mf-library

  Maneja los arquetipos:

  - tmi-mf-lib-commons: Librería que agrupa a las clases y objetos comunes para los distintos proyectos micro-frontends.
  - tmi-mf-lib-services: Librería que agrupa a los servicios compartidos entre microfronts para la comunicación.

- tmi-mf-cross

  Contiene los elementos transversales de la aplicación “LA APLICACIÓN”, por ejemplo, el visor de firmas de la transacción 151 Verificación de firmas.

- tmi-mf-bankdraft-traveler-checks

  Contiene micro-frontend con el dominio tmi-mf-bankdraft-traveler-check el cual contiene la transacción número 151 Verificación de firmas.

- tmi-reverse-proxy

Microservicios

Se inició con la configuración del ambiente de desarrollo, para lo cual se instaló lo siguiente:

- Visual Studio 2022

  El instalador community se descargó de la página web oficial: https://visualstudio.microsoft.com/es/vs/community/

- .NET 6

  Una vez finalizada la descarga se procede con su instalación, una vez terminada ya contamos con el IDE Visual Studio 2022 y el framework .NET 6

Una vez configurado el ambiente de desarrollo, se procedió con la creación de todos los repositorios de fuentes y su desarrollo. A continuación, los repositorios creados para el proyecto:

- TMI-backend-library

  Proyecto de librerías base nuget para ser usados por los servicios Baas Backend as Services y BFF Backend for FrontEnd. Proporcionan funcionalidad de:

  - Seguridad de APIs
  - Almacenamiento en memoria cache
  - Manejo de eventos de dominio
  - Comunicación entre servicios
  - Comunicación con el servicio core
  - Configuración de logs.

- TMI-api-gateway

  Proyecto de API Gateway mediante Ocelot el cual permite centralizar las peticiones https de los micro-frontends y derivarlos a los diferentes servicios backend de los micro-servicios.

- TMI-baas-core

  Proyecto que permite recibir los mensajes de los servicios BackEnd for FrontEnd para ser transformados en una trama xml y ser enviado a la aplicación TX_Server mediante la DLL TX_ClientCom.dll.

- TMI-bff-authentication

  Proyecto que permite la autenticación de los usuarios a la aplicación “LA APLICACIÓN” y obtener información de la sesión generada.

- TMI-baas-authentication

  Proyecto que permite generar y refrescar token, autenticar al usuario y validar claves de red.

- TMI-bff-cross-master-data

  Proyecto que permite a los micro-frontends para obtener información de datos maestros como: monedas u opciones de menú.

- TMI-baas-master-data

Proyecto Backend as Service que permite obtener información de base de datos maestros.

- TMI-bff-trx-151

  Proyecto que se encarga de ejecutar la transacción número 151 Verificación de Firmas.

Convivencia con aplicación existente

Puesto que el producto mínimo viable es la transacción número 151 Verificación de firmas era necesario generar un plan que permita la convivencia entre la versión existente y nueva versión de "LA APLICACIÓN". Para ello, se realizaron las siguientes actividades:

- Creación de COM+ Server.Agent.Wrapper.1.0

  El componente TX_Clientcom.dll es la encargada de enviar las tramas de las transacciones financieras hacia el módulo core de "LA APLICACIÓN", el aplicativo TX_Server.exe. Por ese motivo, este componente era importante seguir utilizando dentro de las micro-servicios.

  Por tanto, para gestionar las instancias del TX_Clientcom.dll tanto para el aplicación monolita y la nueva arquitectura de "LA APLICACIÓN", fue necesario la creación del COM+ Server.Agent.Wrapper.1.0.

  El código fuente se alojó en el repositorio de bitbucket llamado TMI-server-agent-wrapper.

- Creación de COM+ Server.Agent.Authentication.1.0

Este componente COM+ ofrece la posibilidad de comunicación http o https entre la arquitectura monolita y los micro-servicios de la nueva arquitectura de "LA APLICACIÓN".

El código fuente se alojó en el repositorio de bitbucket llamado TMI-server-agent-authentication.

- Creación de ActiveXObject Agente Cliente Client.Agent."LA APLICACIÓN".dll

  Librería encargada de abrir instancia de navegador Google Chrome, en donde se cargará los micro-frontend de las transacciones financieras.

  Esta librería se desarrolló en lenguaje de programación C# y se instaló en todas las estaciones (computadoras) de las tiendas financieras, las cuales son instancias e invocadas desde la aplicación existente.

  El código fuente se alojó en el repositorio de bitbucket llamado TMI-client-agent-"LA APLICACIÓN".

- Modificación de la aplicación monolito de "LA APLICACIÓN"

  A fin de permitir la convivencia entre la aplicación monolito y la arquitectura de micro-frontends, se procedió a modificar archivos de la aplicación web existente. Se modificaron los siguientes archivos:

  menuTxn.htm: Se agregó una nueva función que instancia la dll TMI.Client.Agent."LA APLICACIÓN".dll el cual permite abrir una instancia del navegador Google Chrome con el micro-frontend de la transacción seleccionada en el menú.

enableRemote.asp: Se modificó el método instSignOn encargado de la autenticación de la aplicación web "LA APLICACIÓN", a fin de invocar al servicio tmi-baas-authentication para:

- Generar token api al usuario que se autenticó correctamente en la aplicación monolito.
- Guardar en memoria cache de la nueva arquitectura datos del usuario autenticado.

En el anexo N^0 11, se muestra la aplicación "LA APLICACIÓN" culminado su etapa de desarrollo.

Pruebas

A fin de verificar el correcto funcionamiento del micro-frontend de la transacción 151 Verificación de Firmas, se realizó pruebas funcioales, pruebas automatizadas y de estrés. En el anexo N^0 12 se evidencia la matriz de casos de pruebas funcionales de la transacción 151, en el anexo N^0 13 muestra las pruebas automatizadas realizadas en Cypress.

Implementación

Marcha Blanca

Para la etapa de marcha blanca, se decidió que participarían cuatro tiendas financieras. Los representantes financieros de aquellas tiendas tendrían acceso al nuevo frontend de "LA APLICACIÓN", mediante la aplicación web monolito existente.

Para lo cual se generó diferentes solicitudes de requerimientos de atención (SRA), solicitud de requerimiento de tarea (SRT) y ordenes de cambios (OC) para realizar la implementación en ambiente de producción. A continuación, las actividades realizadas:

- Creación de certificados para IIS

  Solicitud: 3084510

  Fecha de Ejecución: Miércoles 4 de mayo del 2022

  Descripción: Se solicitó la generación de certificado para servidor web IIS en el cual se alojarán las APIS BackEnd for FrontEnd y Baas as a Services.

- Instalación de certificados en IIS

  Solicitud: 3084513

  Fecha de Ejecución: Miércoles 4 de mayo del 2022

  Descripción: Se solicitó la configuración e instalación del certificado generado mediante la solicitud 3084510 en IIS de servidor On-premise del ambiente producción.

- Despliegue de componente DLL en computadoras de las tiendas financieras

  SRA: SRA_2022-00232

  OC: 103247

  Fecha de Ejecución: Martes 10 de Mayo del 2022

  Descripción: Se solicitó la instalación de componente DLL TMI.Client.Agent.”LA APLICACIÓN”.dll a las computadoras de las tiendas financieras involucradas en la etapa de marcha blanca. Este componente

permitió abrir instancias de navegadores Google Chrome desde navegador Internet Explorer.

- Implementación de base de datos listas maestras

  SRT: SRT_2022-01697

  OC: 103251

  Fecha de Ejecución: Martes 10 de mayo del 2022

  Descripción: Se realizó la creación de las tablas de la base de datos de listas maestros y se creó Job de Control-M que permitirá diariamente leer los datos de archivos XML's hacia la base de datos de listas maestras.

- Despliegue cloud de micro-frontend

  SRT: SRT_2022-02029

  OC: 103249

  Fecha de Ejecución: Jueves 9 de Junio del 2022

  Descripción: Se realizó el despliegue en Azure Kubernetes Services de la solución de proxy reverso NGNIX y micro-frontend de la transacción número 151 Verificación de Firmas.

- Despliegue On-Premise de micro-servicios

  SRT: SRT_2022-01859

  OC: 103244

  Fecha de Ejecución: Viernes 10 de junio del 2022

  Descripción: Se realizó el despliegue de las API's que permitirán la ejecución del backend del micro-frontend de transacción número 151 – Verificación de

Firmas. El despliegue solo se realizó a un servidor de producción al cual acceden solo las tiendas involucradas en la etapa de marcha blanca.

- Despliegue On-Premise de web monolito

  SRT: SRT_2022-01860

  OC: 103246

  Fecha de Ejecución: Miércoles 22 de junio del 2022

  Descripción: Se realizó el despliegue de archivos de la aplicación web monolito que permite realizar la convivencia con los micro-frontend. En este caso permitió habilitar el nuevo micro-frontend de la transacción 151 Verificación de firmas a las tiendas correspondientes a las tiendas financieras. En el anexo N^{0} 14 se muestra la conformidad de pase.

Pase a nivel Nacional

Una vez culminado la etapa de marcha blanca y validar el correcto funcionamiento de la arquitectura micro-frontend, se procedió a realizar el pase de la nueva arquitectura micro-frontend para todas las tiendas financieras del cliente “EL CLIENTE” a nivel nacional. A continuación, las actividades realizadas:

- Creación e instalación de certificados para IIS

  Solicitudes: 3123364 – 3123370 – 3123377 – 3123381 – 3123382 – 3123386 – 3123392 – 3123395 – 3123398 - 3123407

  Descripción: Se solicitó la generación e instalación de certificados para servidor web IIS en el cual se alojarán las APIS BackEnd. Se tuvo que generar una solicitud por cada servidor IIS.

- Despliegue de componente DLL en computadoras de las tiendas financieras

- Implementación de base de datos listas maestras

  SRT: SRT_2022-04003

  OC: 106539

  Fecha de Ejecución: Lunes 1 de agosto del 2022

  Descripción: Se realizó implementación de base de datos ibtmipmaestros al servidor de base de datos AlwaysOn que es utilizado por la malla de servidores de aplicaciones y se agregó en servidor AlwaysOn en la lista de servidores del Job Control-M que se encargaría de migrar datos maestros de archivos XML a base de datos.

- Despliegue cloud de micro-frontend

  SRT: SRT_2022-03808

  OC: 106161

  Fecha de Ejecución: Viernes 22 de Julio del 2022

  Descripción: Se realizó el despliegue de actualización de micro-frontend de transacción 151 Verificación de firmas en donde se mejoró el look and feel.

- Despliegue On-Premise de micro-servicios

  SRT: SRT_2022-03733

  OC: 106062

  Fecha de Ejecución: Viernes 22 de julio del 2022

Descripción: Se realizó el despliegue de API's On-premise a toda la malla de servidores nodos de "LA APLICACIÓN" que permitirán la ejecución del backend del micro-frontend de transacción número 151 Verificación de Firmas.

- Despliegue On-Premise de web monolito

  SRT: SRT_2022-03734

  OC: 106274

  Fecha de Ejecución: Martes 2 de agosto del 2022

  Descripción: Se realizó el despliegue de archivos de la aplicación web monolito que permitiría realizar la convivencia con los micro-frontend. Permitiría habilitar el nuevo micro-frontend de la transacción 151 Verificación de firmas a todas las tiendas financieras a nivel nacional. En el anexo N^0 15 se muestra el correo de conformidad del pase a nivel nacional.

CAPÍTULO IV. RESULTADOS

En el presente capítulo de la tesis se presentan los resultados de los objetivos específicos planteando en el inicio del proyecto.

El primer objetivo específico trazado fue realizar el cambio progresivo equivalente mediante la arquitectura micro-frotend para reducir la obsolescencia de software de la aplicación web financiera.

Para validar el cambio progresivo se recabo información acerca de la cantidad de usuarios, representantes financieros, que tuvieron acceso a la nueva arquitectura micro-frontend de "LA APLICACIÓN" durante la implementación a ambiente productivo.

Tabla 2 Aumento de uso de micro-frontend de transacción 151 en Google Chrome

Fecha	Usuarios en Google Chrome
9-Jun	0
16-Jun	0
23-Jun	45
30-Jun	45
7-Jul	45
14-Jul	45
21-Jul	45
28-Jul	45
4-Ago	1879
11-Ago	1879

Fuente: Elaboración propia

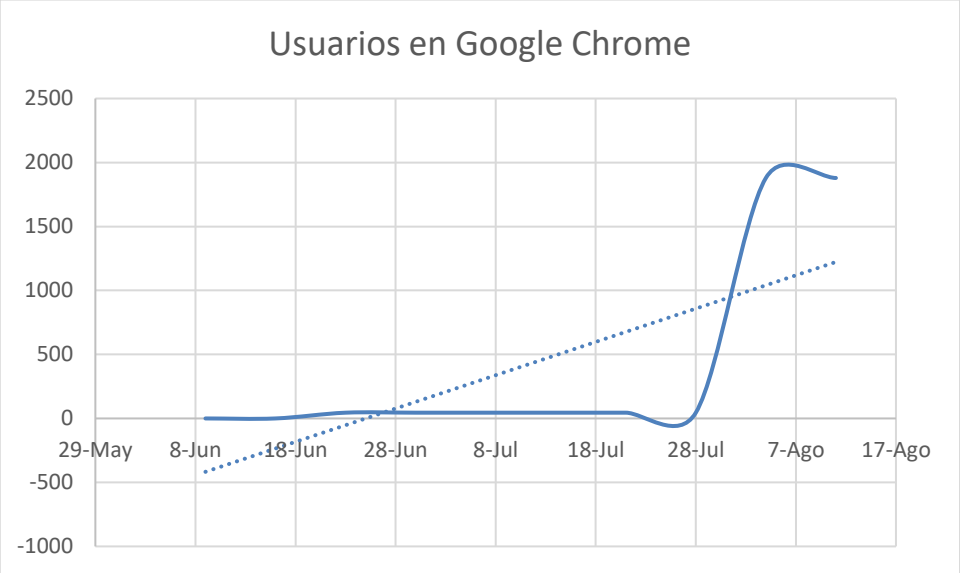


Figura 6 Aumento de usuarios usando micro-frontend transacción 151 en Google Chrome

Fuente: Elaboración propia

Según el análisis de los resultados obtenido en la tabla 2 y figura 6, podemos afirmar que el cambio a la nueva arquitectura fue progresivo. Antes de la fase de marcha blanca, ningún usuario tuvo acceso a la nueva aplicación. En la fase de marcha blanca, 45 representantes financieros utilizaron el micro-frontend de la transacción 151 en el navegador Google Chrome. Finalmente, una vez realizado el pase a nivel nacional, a partir del 4 de agosto 1879 usuarios tuvieron acceso a la nueva arquitectura micro-frontend de “LA APLICACIÓN”. Asimismo, en la figura 6 se identifica que el cambio a la nueva arquitectura micro-frontend tuvo una tendencia ascendente y progresiva en el tiempo.

El segundo objetivo específico definido fue reducir el riesgo tecnológico mediante la arquitectura micro-frontend para la obsolescencia de software de la aplicación fue financiera. Para validar el cumplimiento del presente objetivo, se recolectó información del uso del navegador Internet Explorer 11, el cual es obsoleto, frente al uso del navegador Google Chrome para acceder a la transacción 151 Verificación de Firmas.

Tabla 3 *Reducción de riesgo tecnológico mediante micro-frontend de transacción 151 en 2022*

Fecha	Usuarios en Internet Explorer	Usuarios en Google Chrome
9-Jun	1879	0
16-Jun	1879	0
23-Jun	1834	45
30-Jun	1834	45
7-Jul	1834	45
14-Jul	1834	45
21-Jul	1834	45
28-Jul	1834	45
4-Ago	0	1879
11-Ago	0	1879

Fuente: Elaboración propia

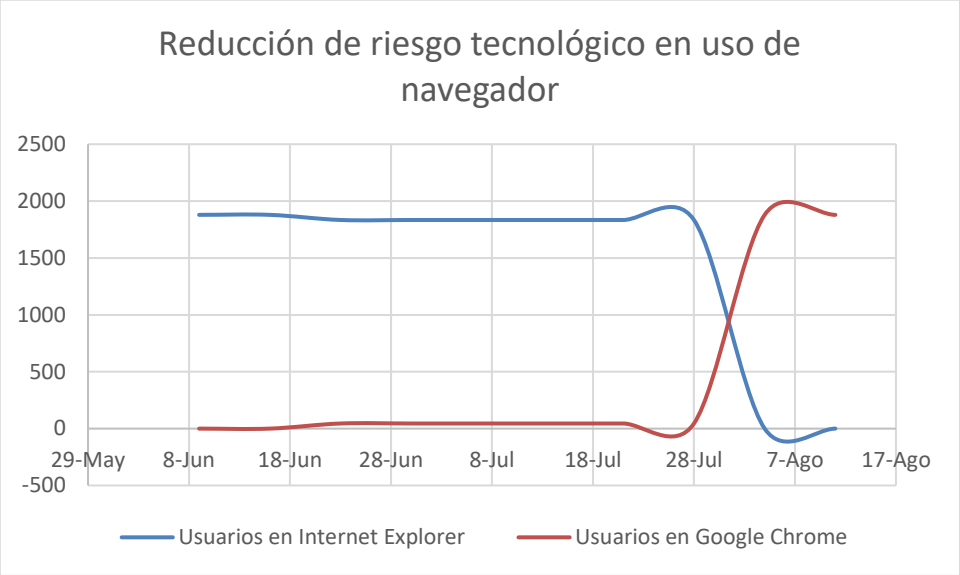


Figura 7 Uso progresivo de micro-frontend transacción 151

Fuente: Elaboración propia

De acuerdo con el análisis de los resultados obtenidos en la tabla 3 y figura 7, al inicio del proyecto el 100% de los usuarios de “LA APLICACIÓN” ingresaban a la transacción 151 verificación de firmas mediante el navegador Internet Explorer 11. Luego, durante la fase de marcha blanca se redujo en un 2.40% el riesgo tecnológico. Finalmente, una vez realizado el pase a ambiente productivo a nivel nacional, se logró reducir en un 100% el riesgo tecnológico de “LA APLICACIÓN” en la transacción 151 Verificación de Firmas.

El tercer objetivo específico establecido fue la refactorización de la aplicación monolito mediante la arquitectura micro-frontend para la obsolescencia de software de la aplicación web financiera.

Para fundamentar el cumplimiento del presente objetivo, se recopilo información acerca de la cantidad de pruebas unitarias de “LA APLICACIÓN” entre la arquitectura monolito y micro-frontend.

Tabla 4 Cantidad de pruebas unitarias en arquitectura micro-frontend de "LA APLICACIÓN"

Proyecto	Número de Pruebas Unitarias
TMI-backend-library	50
TMI-baas-core	154
TMI-bff-authentication	1
TMI-baas-authentication	12
TMI-bff-cross-master-data	15
TMI-baas-master-data	20
TMI-bff-trx-151	4
tmi-mf-base	119
tmi-mf-library	93
tmi-mf-cross	121
tmi-mf-bankdraft-traveler-checks	78
Total	667

Fuente: Elaboración propia

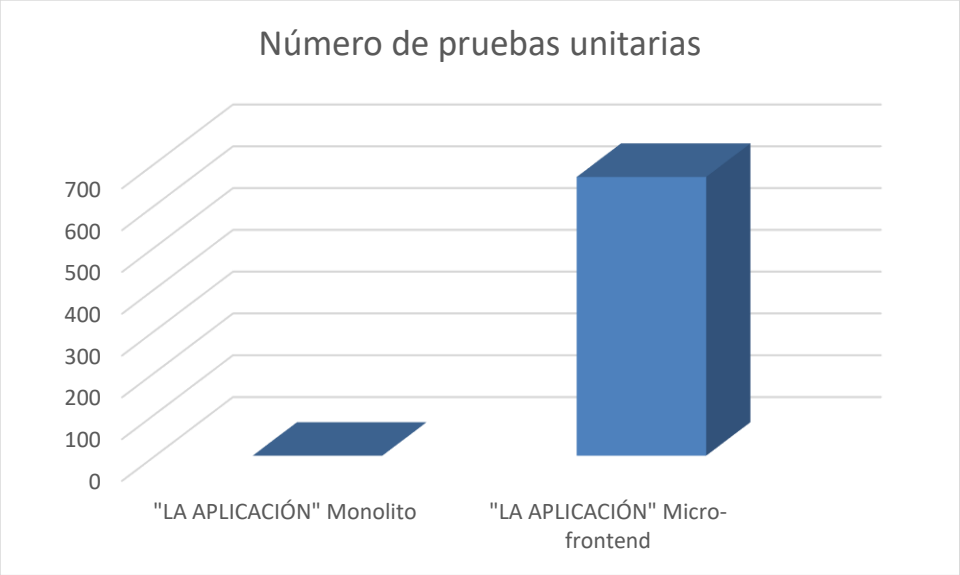


Figura 8 Número de pruebas unitarias por tipo de arquitectura

Fuente: Elaboración propia

De acuerdo con la tabla 4 y figura 6, se evidencia que en la arquitectura monolito de "LA APLICACIÓN" no contaba con ninguna prueba unitaria, en cambio en la arquitectura micro-frontend, se implementó un total de 667 pruebas unitarias. De esta manera, se cumplió en un 100% la refactorización de la aplicación monolito.

CAPÍTULO V. CONCLUSIONES Y RECOMENDACIONES

Con la presente tesis se describió la experiencia profesional realizada en el cliente bancario de TS NET S.A (EVOL), en donde se implementó la arquitectura micro-frontend lográndose cumplir los objetivos propuestos.

Con respecto al primer objetivo, se verificó que existió una mejora progresiva en la transacción 151 Verificación de firmas mediante implementación la arquitectura micro-frontend, en donde se logró la migración de 100% de la transacción sin afectar otras transacciones de "LA APLICACIÓN", en comparación del antecedente *Migración de infraestructura del chatbot para mitigar el riesgo de seguridad y obsolescencia en una entidad financiera en el 2021* de Amaro Proa (2021) en donde también se logró la migración de infraestructura. Por tanto, se concluyó que hubo una mejora resaltante pues una vez implementado el presente proyecto, el total de representantes financieros salieron de obsolescencia de software dejando de utilizar una web con arquitectura monolito y contando una aplicación web en una arquitectura moderna.

Por otro lado, con respecto al segundo objetivo se concluyó que hubo una mejora del 100% en reducir el riesgo tecnológico mediante la arquitectura micro-frontend en la transacción 151 Verificación de firmas de "LA APLICACIÓN" en donde se eliminó el uso del navegador Internet Explorer 11 y pasó a usar el navegado Google Chrome. Como indicó el antecedente *Análisis de la obsolescencia de software por el avance de la tecnología* de Vélez Aguirre (2019), si se mantenía la arquitectura monolito y el navegador Internet Explorer 11, provocaría lentitud y problemas de seguridad que afectaría al usuario; sin embargo, se concluyó que se tuvo una mejora al tener un 100% de representantes financieros usando la transacción 151 en una arquitectura y navegador migrado.

Finalmente, en cuanto al tercer objetivo de refactorización de la aplicación monolito mediante la arquitectura micro-frontend para la obsolescencia de software de la aplicación web financiera, se concluyó que se logró 100% de la refactorización, pues se migró correctamente la arquitectura; además, que a diferencia de la arquitectura monolito en la arquitectura micro-frontend se implementó 667 pruebas unitarias evitando errores en la aplicación.

RECOMENDACIONES

A continuación, se presentan las recomendaciones. En primer lugar, se recomienda utilizar servidores en la nube para almacenar los micro-servicios de los micro-frontends; pues actualmente, se cuenta con servidores OnPremise. En segundo lugar, la implementación de micro-frontend se puede aplicar en todos los módulos o transacciones de la aplicación web financiera “LA APLICACIÓN”. En tercer lugar, se aconseja realizar pruebas de la aplicación web con arquitectura micro-frontend en navegadores diferentes de Google Chrome, como Firefox o Brave a fin de encontrar un navegador más ligero y que consuma menos recursos de la estación de trabajo. Por último, considerar la presente investigación como base para futuras investigaciones e implementaciones en otras entidades financieras.

REFERENCIAS

Adservio team. (1 de Abril de 2022). *Top Micro Frontend Frameworks*. Recuperado el 7 de Julio de 2022, de Software Development & IT Service Provider Adservio: https://www.adservio.fr/post/top-micro-frontend-frameworks

Amaro Proa, S. J. (2021). Migración de infraestructura del chatbot para mitigar el riesgo de seguridad y obsolescencia en una entidad financiera en el 2021. *(Tesis).* Universidad Nacional Mayor de San Marcos, Lima, Perú. doi:https://hdl.handle.net/20.500.12672/17835

Anderson, R., Latham, L., Larkin, K., smandia, Muradian, A., Addie, S., . . . Roth, D. (3 de Junio de 2022). *Distributed caching in ASP.NET Core*. Recuperado el 21 de Agosto de 2022, de Microsoft Docs: https://docs.microsoft.com/en-us/aspnet/core/performance/caching/distributed?view=aspnetcore-6.0

Angular Components Team. (s.f.). *Angular Material*. Recuperado el 14 de Agosto de 2022, de Angular Material: https://material.angular.io/

Atlassian. (s.f.). *¿Para qué se utiliza Jira Software?* Recuperado el 14 de Agosto de 2022, de Atlassian: https://www.atlassian.com/es/software/jira/guides/use-cases/what-is-jira-used-for

Atlassian. (s.f.). *Breve presentación de BitBucket*. Recuperado el 14 de Agosto de 2022, de Bitbucket: https://bitbucket.org/product/es/guides/getting-started/overview

Atuesta Maestre, J. A., & Nariño Ciro, J. F. (2020). Modelo de Arquitectura para Front-End (Basado en Micro-Frontends) Aplicado al Producto Digital de Fuerza Especializada de Vivienda del Banco de Bogotá. *(Trabajo de Grado).* Universidad Distrital Francisco José de Caldas, Bogotá, Colombia. Obtenido de https://repository.udistrital.edu.co/handle/11349/25071?show=full

Bartels, B., Ermel, U., Pecht, M., & Sandborn, P. (2012). *Strategies to the Prediction, Mitigation and Management of Product Obsolescence.* Hoboken, New Jersey, Canadá: John Wiley & Sons, Inc. Obtenido de https://www.perlego.com/book/1014674/strategies-to-the-prediction-mitigation-and-management-of-product-obsolescence-pdf

Chiaretta, S. (2018). *Front-end Development with ASP.NET Core, Angular, and Bootstrap.* John Wiley & Sons, Inc.

Clarisó, R., & Cabot, J. (1 de Febrero de 2018). El software también envejece. *EL PAIS*. Obtenido de https://elpais.com/tecnologia/2018/01/16/actualidad/1516121401_534975.html

Deloitte. (s.f.). *Arquitectura Microfrontends | Deloitte España*. Recuperado el 11 de Julio de 2022, de Deloitte Spain: https://www2.deloitte.com/es/es/pages/technology/articles/arquitectura-microfrontends.html

Dueñas Ramírez, L. M., & Villegas López, G. A. (2020). La gestión de activos y la obsolescencia tecnológica en el análisis del ciclo de vida. *XXII Congreso Internacional de Mantenimiento y Gestión de Activos.* Bogotá. doi:10.13140/RG.2.2.10179.22563

Erinç, Y. K. (20 de Agosto de 2020). *The SOLID principles of Object-Oriented programming explained in plain english*. Recuperado el 21 de Agosto de 2022, de freeCodeCamp.org: https://www.freecodecamp.org/news/solid-principles-explained-in-plain-english/

García Chihuanhuaylla, M. (2022). Implementación de una arquitectura micro-frontend para optimizar el desarrollo y despliegue de la aplicación web del sistema de información universitaria de la SUNEDU. *(Tesis).* Universidad Nacional Mayor de San Marcos, Lima, Perú. Obtenido de https://hdl.handle.net/20.500.12672/17887

Gartner. (6 de Abril de 2022). *Gartner Forecasts Worldwide IT Spending to Reach $4.4 Trillion in 2022* . Obtenido de www.gartner.com: https://www.gartner.com/en/newsroom/press-releases/2022-04-06-gartner-forecasts-worldwide-it-spending-to-reach-4-point-four-trillion-in-2022

Geers, M. (2020). *Micro Frontends in action.* Simon and Schuster.

Geers, M. (s.f.). *MicroFrontends: Extendiendo la idea de microservicio al desarrollo frontend*. Recuperado el 30 de 05 de 2022, de https://micro-frontends-es.org/

Google. (s.f.). *Angular*. Recuperado el 14 de Agosto de 2022, de Angular.lat: https://docs.angular.lat/docs

Google. (s.f.). *Navegador web Google Chrome*. Recuperado el 13 de Octubre de 2022, de Google.com: https://www.google.com/intl/es/chrome/

Groner, L. (2014). *Learning javaScript data structures and algorithms.* Birmingham: Packt Publishing, Limited.

Gutiérrez Gonzáles, Á., & López Goytia, J. (2016). *Desarrollo y programación en entornos web* (1 ed.). Ciudad de México, México: Alfaomega. Obtenido de https://www.alphaeditorialcloud.com/reader/desarrollo-y-programacion-en-entornos-web

Hughes, A. (17 de Junio de 2019). *Microsoft SQL Server*. Recuperado el 20 de Agosto de 2022, de SearchDataManagement: https://www.techtarget.com/searchdatamanagement/definition/SQL-Server

IBM. (14 de Abril de 2021). *Applets y aplicaciones Java*. Obtenido de Ibm.com: https://www.ibm.com/docs/es/i/7.2?topic=platform-java-applets-applications

Invision. (s.f.). *InVision for Design Teams*. Recuperado el 14 de Agosto de 2022, de invisionapp: https://www.invisionapp.com/teams/design

JFrog Ltd. (7 de Noviembre de 2016). *Artifactory - universal artifact repository manager*. Recuperado el 13 de Octubre de 2022, de JFrog: https://jfrog.com/artifactory/

King, B. (20 de Marzo de 2015). *Procedimientos recomendados en c# - peligros de la violación de los principios SOLID en c#*. Obtenido de Microsoft Docs: Procedimientos recomendados en c# - peligros de la violación de los principios SOLID en c#

Krishnamurthy, S. (Octubre - Noviembre de 2019). What are micro frontends. (H. Mohan, Ed.) *IEEE India Info, 14*(4), 105 - 109. Obtenido de https://site.ieee.org/indiacouncil/files/2019/12/p105-p109.pdf

Latouche, S. (2014). *Hecho para tirar: la irracionalidad de la obsolescencia programada.* Barcelona, España: Ediciones Octaedro, S.L. Obtenido de https://elibro.bibliotecaupn.elogim.com/es/lc/upnorte/titulos/116819

Microsoft. (s.f.). *[MS-WPO]: Internet Information Services*. Recuperado el 13 de Octubre de 2022, de Microsoft.com: https://learn.microsoft.com/en-us/openspecs/windows_protocols/ms-wpo/5e6169bc-e287-4384-bce0-68980cf2a491

Microsoft. (s.f.). *¿Qué es Azure?* Recuperado el 13 de Octubre de 2022, de Microsoft.com: https://learn.microsoft.com/es-es/training/modules/intro-to-azure-fundamentals/what-is-microsoft-azure?ns-enrollment-type=learningpath&ns-enrollment-id=learn.az-900-describe-cloud-concepts

Microsoft. (6 de Junio de 2017). *ASP Overview*. Obtenido de Docs.microsoft.com: https://docs.microsoft.com/en-us/previous-versions/iis/6.0-sdk/ms524929(v=vs.90)

Microsoft. (24 de Junio de 2022). *Finalización del soporte de la aplicación de escritorio Internet Explorer 11 para ciertos sistemas operativos*. Obtenido de Microsoft.com: https://docs.microsoft.com/es-es/lifecycle/announcements/internet-explorer-11-end-of-support

Microsoft. (s.f.). *Documentation for Visual Studio Code*. Recuperado el 14 de Agosto de 2022, de Visualstudio.com: https://code.visualstudio.com/docs

Microsoft. (s.f.). *Entity framework documentation*. Recuperado el 14 de Agosto de 2022, de Microsoft.com: https://docs.microsoft.com/en-us/ef/

Microsoft. (s.f.). *Frequently asked questions (FAQ) about Edge in the enterprise*. Recuperado el 17 de Julio de 2022, de Microsoft.com: https://docs.microsoft.com/en-us/deployedge/faqs-edge-in-the-enterprise

Microsoft. (s.f.). *JavaScript with syntax for types*. Recuperado el 14 de Agosto de 2022, de typescriptlang: https://www.typescriptlang.org/

Microsoft. (s.f.). *Usar controles ActiveX en Internet Explorer 11*. Recuperado el 10 de Agosto de 2022, de Microsoft: https://support.microsoft.com/es-es/windows/usar-controles-activex-en-internet-explorer-11-25738d05-d357-39b4-eb2f-

fdd074bbf347#:~:text=Los%20controles%20ActiveX%20son%20peque%C3%B1as,cotizaciones%20mientras%20navegas%20por%20Internet.

Microsoft. (s.f.). *Visual Studio 2022*. Recuperado el 14 de Agosto de 2022, de Visual Studio: https://visualstudio.microsoft.com/es/vs/

Microsoft. (s.f.). *What is .NET?* Recuperado el 14 de Agosto de 2022, de Microsoft.com: https://dotnet.microsoft.com/en-us/learn/dotnet/what-is-dotnet

Monte Galiano, J. L. (2016). *Implantar scrum con éxito* (Primera ed.). Barcelona, España: Editorial UOC. Obtenido de https://elibro.bibliotecaupn.elogim.com/es/lc/upnorte/titulos/58575

Mozilla Corporation's. (21 de Setiembre de 2022). *Microsoft Internet Explorer - MDN Web Docs Glossary: Definitions of Web-related terms | MDN*. Obtenido de MDN Web Docs: https://developer.mozilla.org/en-US/docs/Glossary/Microsoft_Internet_Explorer

Node.js. (s.f.). *Acerca de Node.js*. Recuperado el 13 de Octubre de 2022, de Node.js: https://nodejs.org/es/about/

Oracle. (s.f.). *Applet (java SE 10 & JDK 10)*. Recuperado el 19 de Julio de 2022, de Oracle.com: https://docs.oracle.com/javase/10/docs/api/java/applet/package-summary.html

Peltonen, S., Mezzalira, L., & Taibi, D. (2021). Motivations, benefits, and issues for adopting Micro-Frontends: A Multivocal Literature Review. *Information and Software Technology, 136*(106571). doi:10.48550/ARXIV.2007.00293

Pérez Ibarra, S. G., Quispe, J. R., Mullicundo, F. F., & Lamas, D. A. (2021). Herramientas y tecnologías para el desarrollo web desde el FrontEnd al BackEnd. *XXIII Workshop de Investigadores en Ciencias de la Computación (WICC 2021, Chilecito, La Rioja)*, (págs. 347-350). Recuperado el 28 de 04 de 2022, de http://sedici.unlp.edu.ar/bitstream/handle/10915/120476/Ponencia.pdf-PDFA.pdf?sequence=1&isAllowed=y

PlantUML. (s.f.). *herramienta de código abierto que utiliza descripciones textuales simples para dibujar diagramas UML*. Recuperado el 14 de Agosto de 2022, de PlantUML.com: https://plantuml.com/es/

Postman. (s.f.). *Postman*. Recuperado el 14 de Agosto de 2022, de Postman.com: https://www.postman.com/

Rajagopal, S., Erkoyuncu, J. A., & Roy, R. (2014). Software Obsolescence in Defence. *Procedia CIRP, 22*, 76-80. doi:https://doi.org/10.1016/j.procir.2014.07.121

Real Academia Española. (2022). *Diccionario de la lengua española*. Obtenido de https://dle.rae.es/software

Recio García, J. A. (2016). *HTML5, CSS3 y JQuery: curso práctico.* Madrid, España: RA-MA Editorial. Obtenido de https://elibro.bibliotecaupn.elogim.com/es/ereader/upnorte/106494

Roldán Martínez, D., Valderas Aranda, P. J., & Torres Bosch, V. (2018). *Microservicios: un enfoque integrado.* Madrid, España: RA-MA Editorial. Obtenido de https://elibro.bibliotecaupn.elogim.com/es/lc/upnorte/titulos/106522

Vega Antonio, O. (2012). Efectos colaterales de la obsolescencia tecnoloógica. *Facultad de Ingenieria, 21*(32), 55 - 62. Recuperado el 15 de Mayo de 2022, de https://www.redalyc.org/articulo.oa?id=413940771005

Vélez Aguirre, S. (2019). Análisis de la obsolescencia de software por el avance de la tecnologìa. *Trabajo de titulación.* Universidad Estatal de Milagro, Milagro, Ecuador. Obtenido de http://repositorio.unemi.edu.ec/handle/123456789/4453

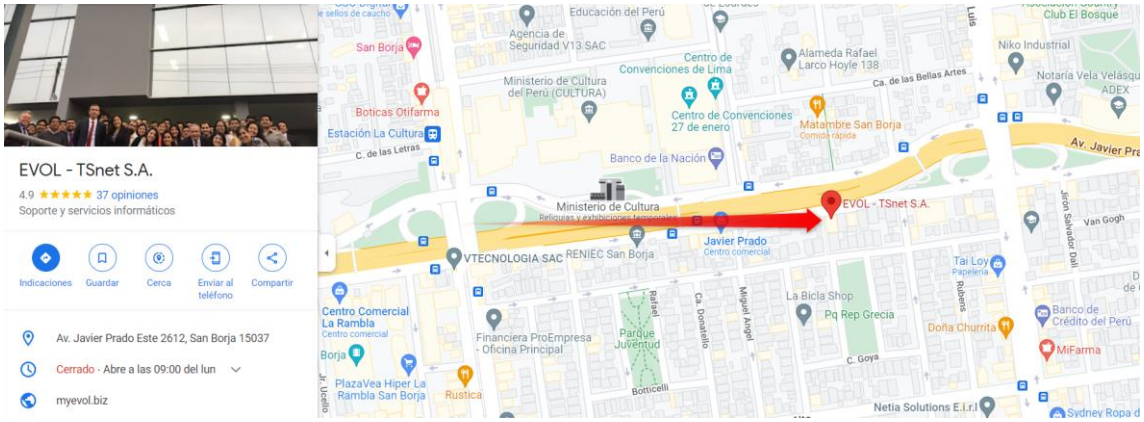
Wagner, B., Dykstra, T., M. Santos, R., Sharkey, K., Coulter, D., Newcomb, B., . . . Onderka, P. (8 de Julio de 2022). *Un paseo por c#: Información general*. Recuperado el 14 de Agosto de 2022, de Microsoft Docs: https://docs.microsoft.com/es-es/dotnet/csharp/tour-of-csharp/

Yang, C., Liu, C., & Su, Z. (2019). Research and application of micro frontends. *IOP conference series: materials science and engineering, 490*(062082), 6. doi:10.1088/1757-899X/490/6/062082

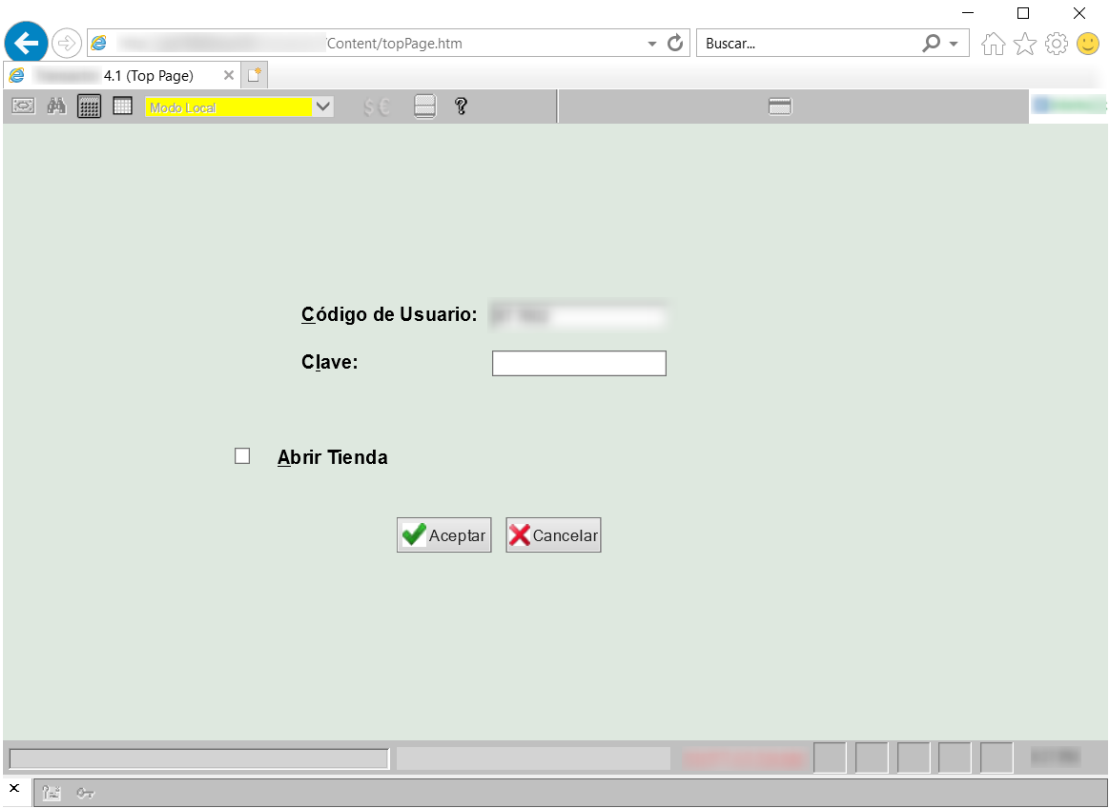
Zofío Jiménez, J. (2013). *Aplicaciones web.* Madrid, España: Macmillan Iberia, S.A. Obtenido de https://elibro.bibliotecaupn.elogim.com/es/lc/upnorte/titulos/43262

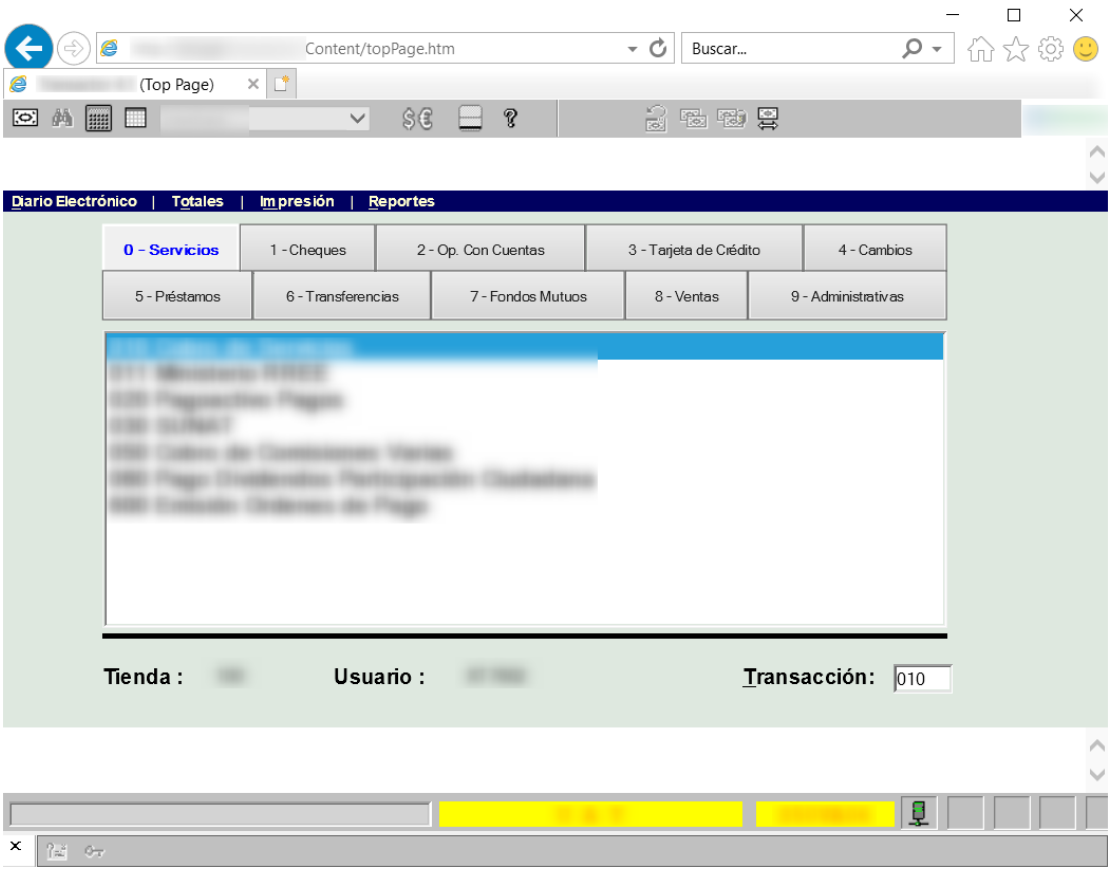
ANEXOS

ANEXO N.° 1. Ubicación geográfica de sede en Perú de TS NET S.A

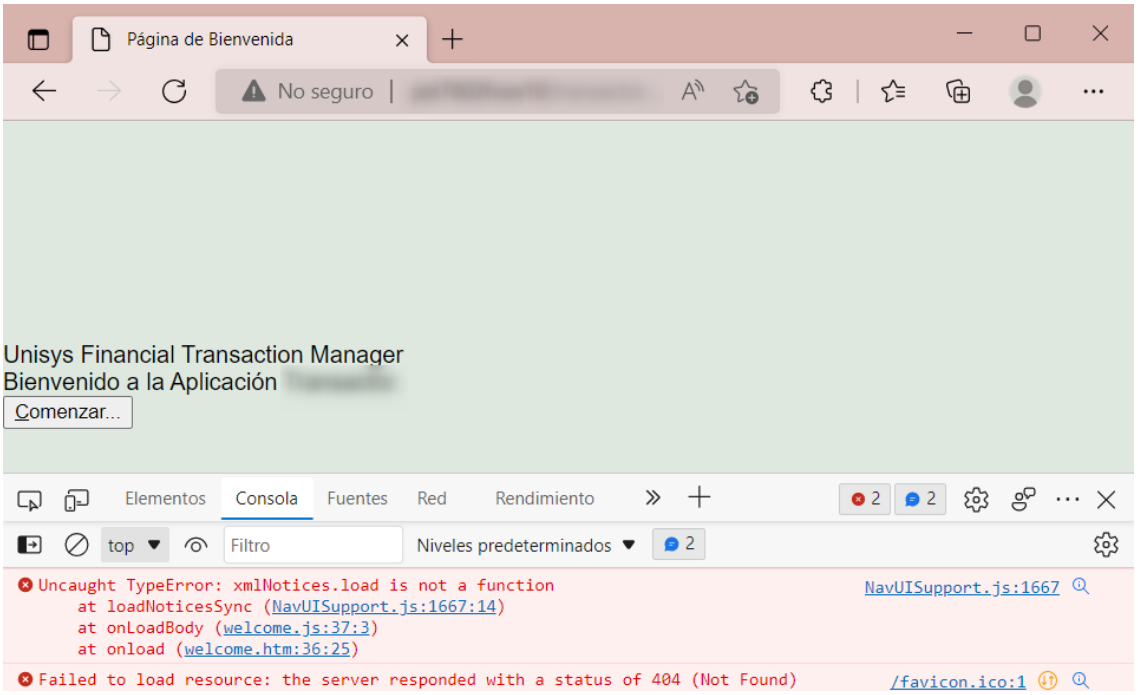


ANEXO N.° 2. "LA APLICACIÓN" Página de acceso y menú

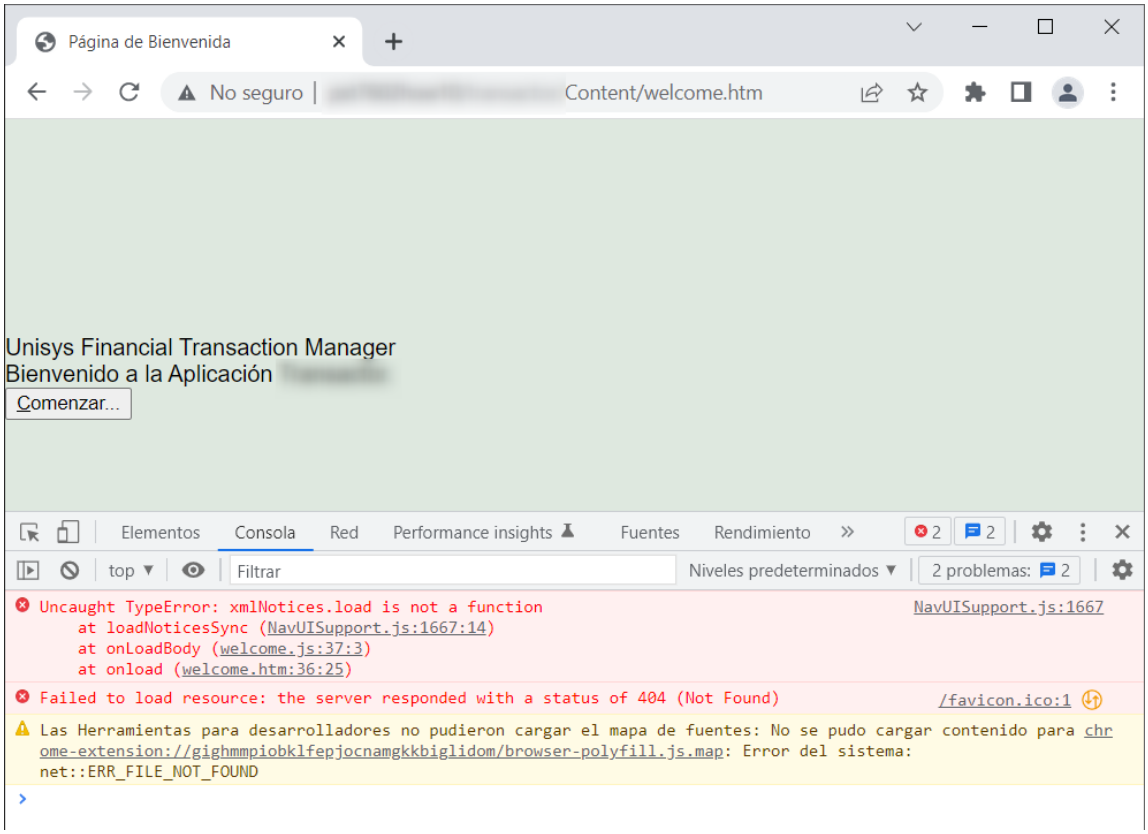




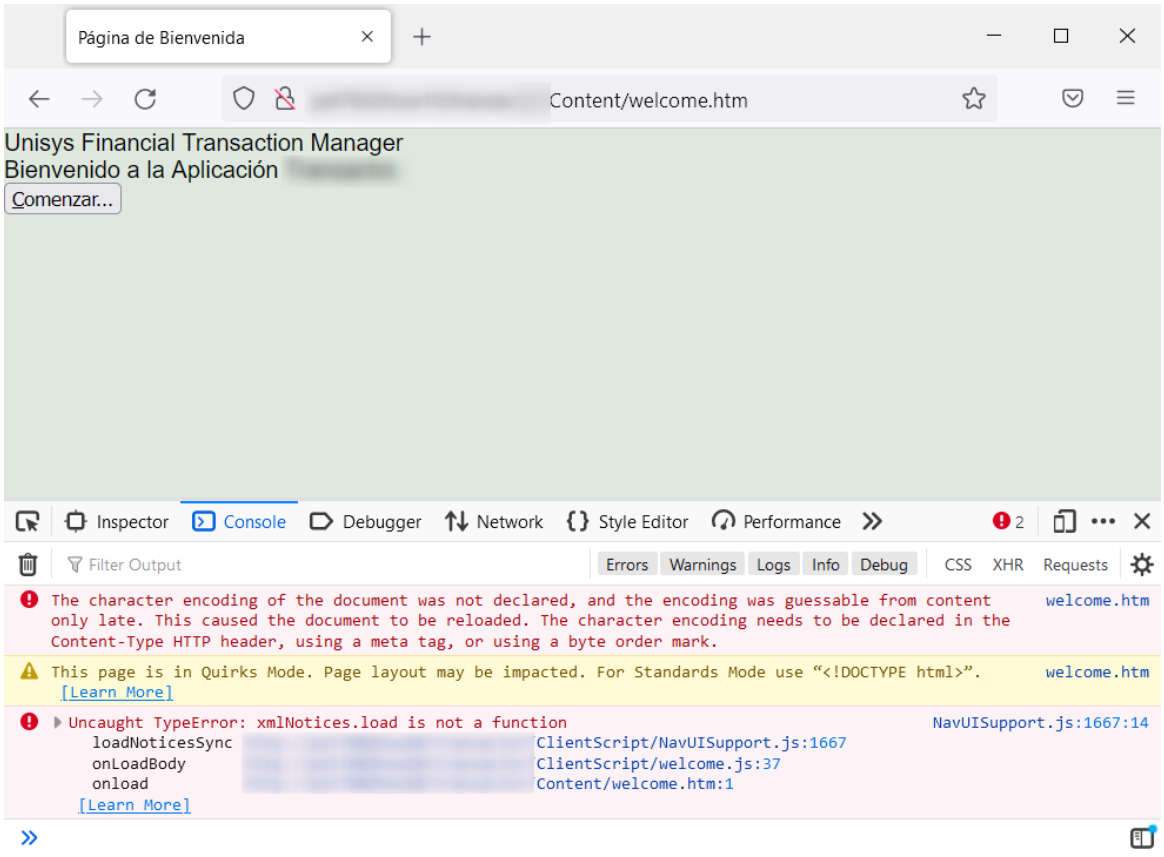
ANEXO N.° 3. Errores de "LA APLICACIÓN" en navegador Microsoft Edge



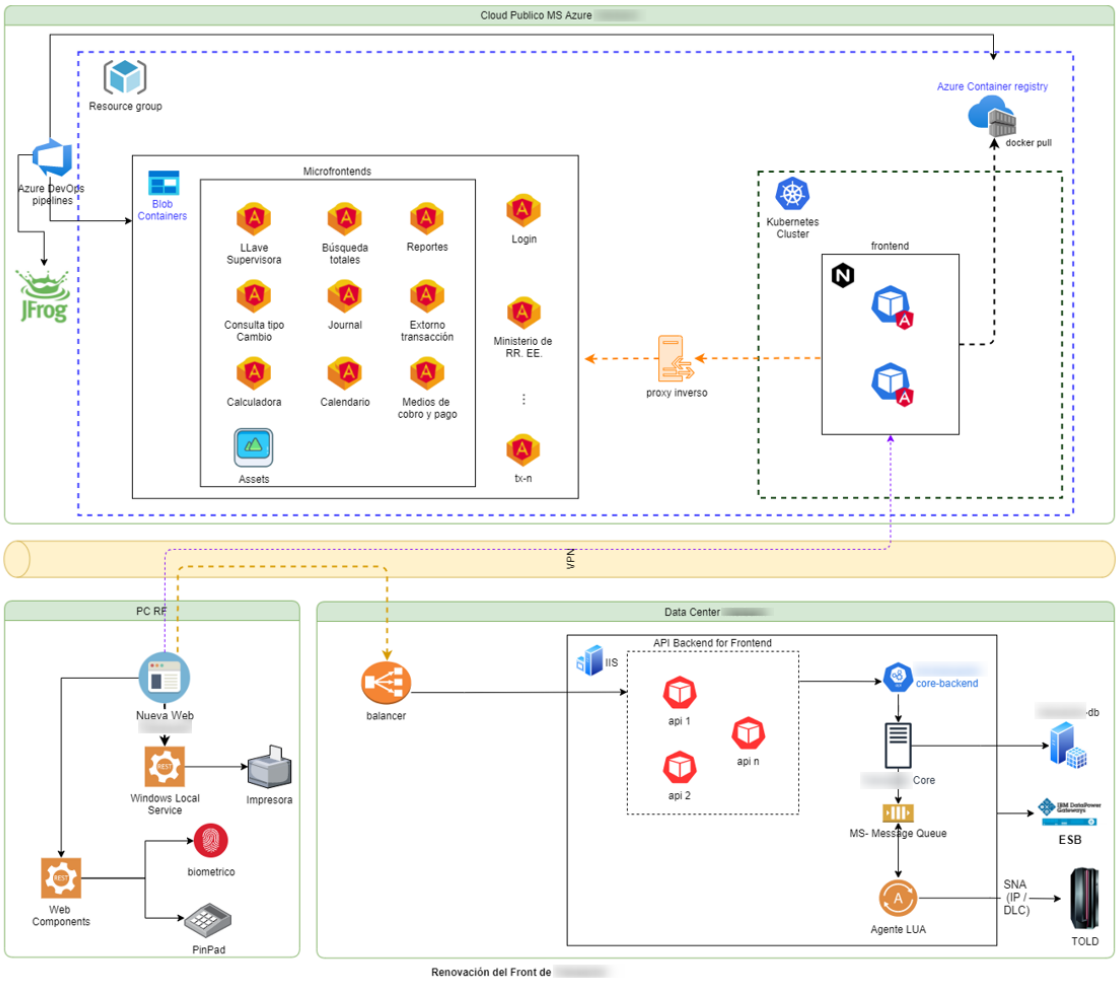
ANEXO N.° 4. Errores de "LA APLICACIÓN" en navegador Google Chrome



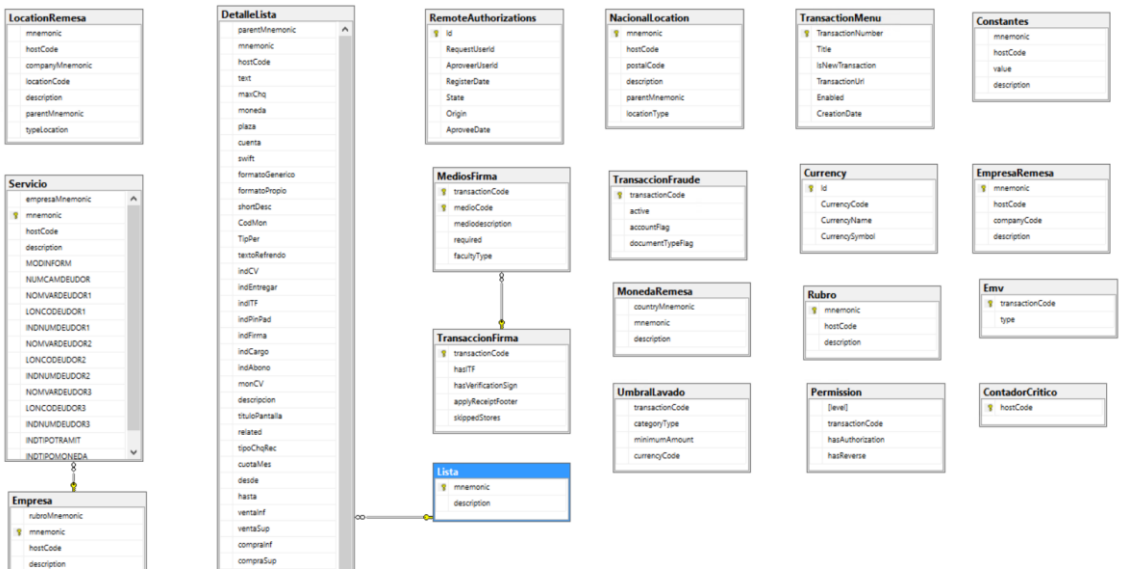
ANEXO N.° 5. Errores de “LA APLICACIÓN” en navegador Firefox



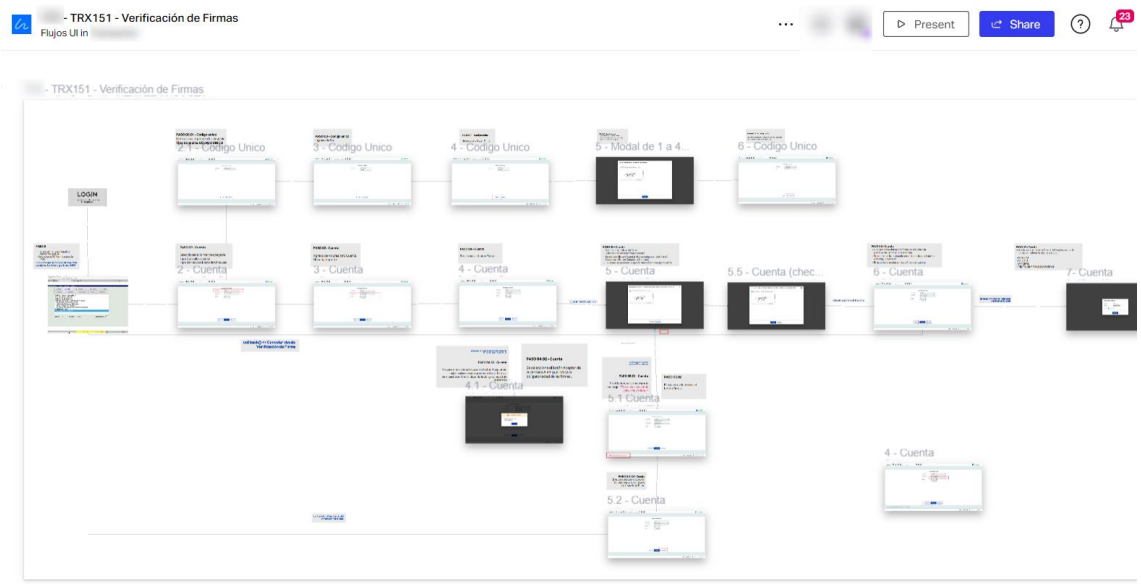
ANEXO N.° 6. Arquitectura propuesta para “LA APLICACIÓN”.



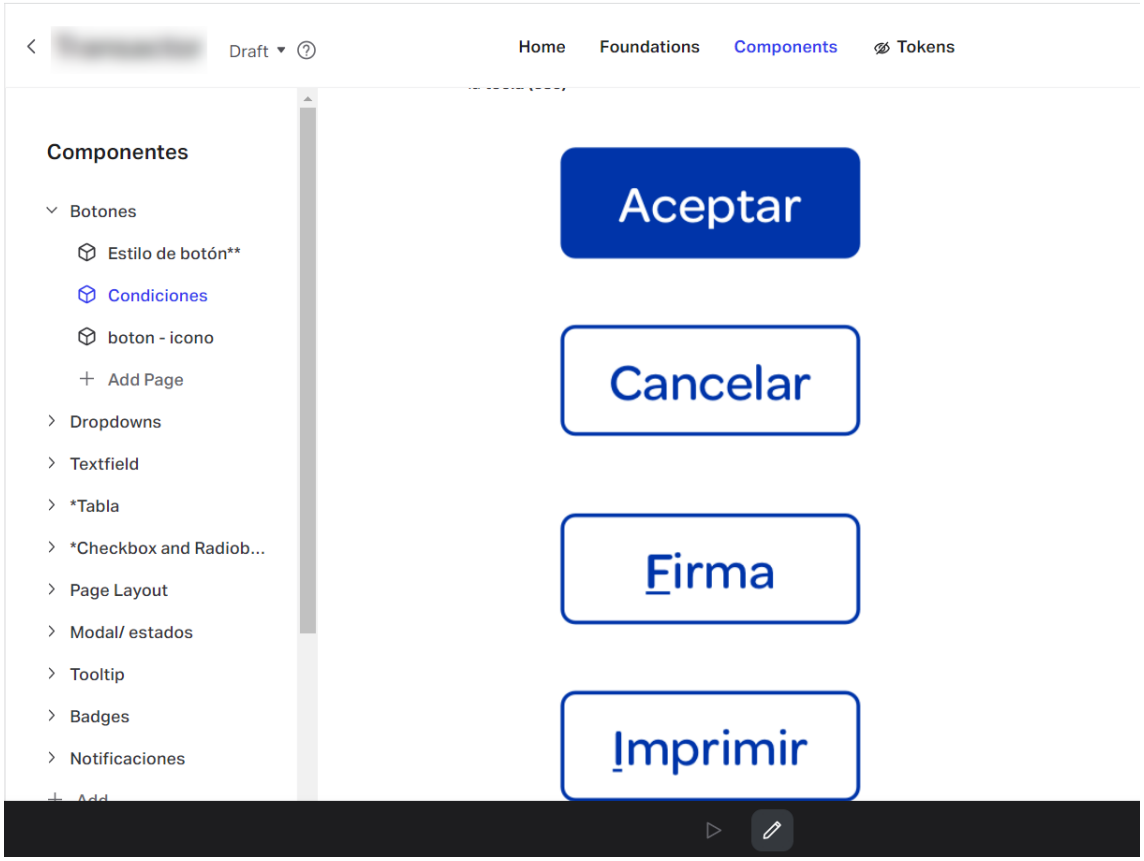
ANEXO N.° 7. Diagrama de base de datos maestros.

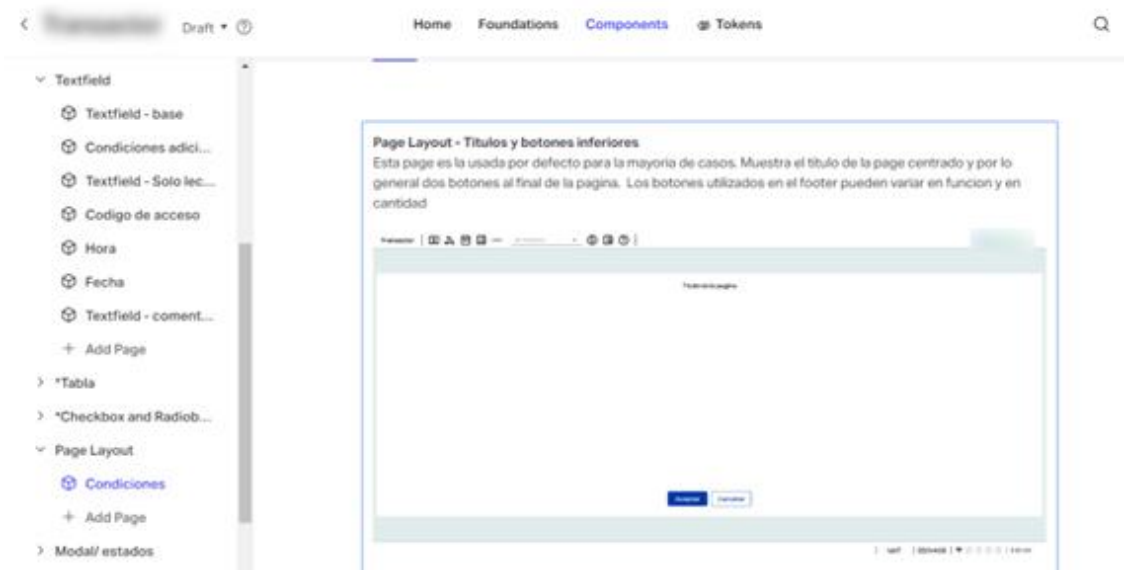


ANEXO N.° 8. Diseño UX de transacción 151 de “LA APLICACIÓN”.

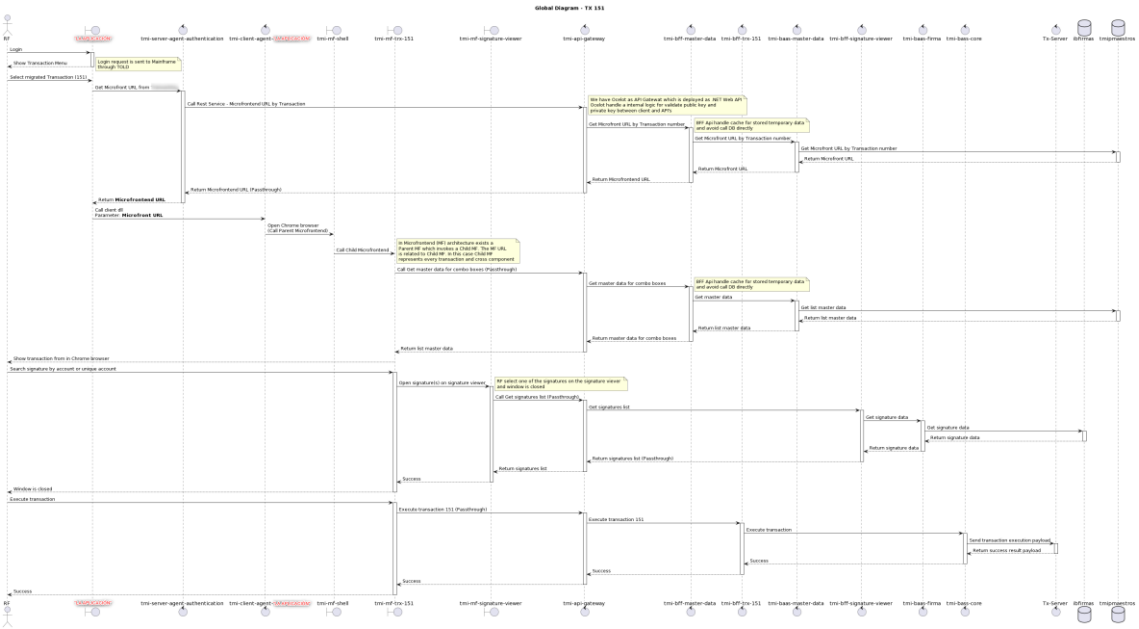


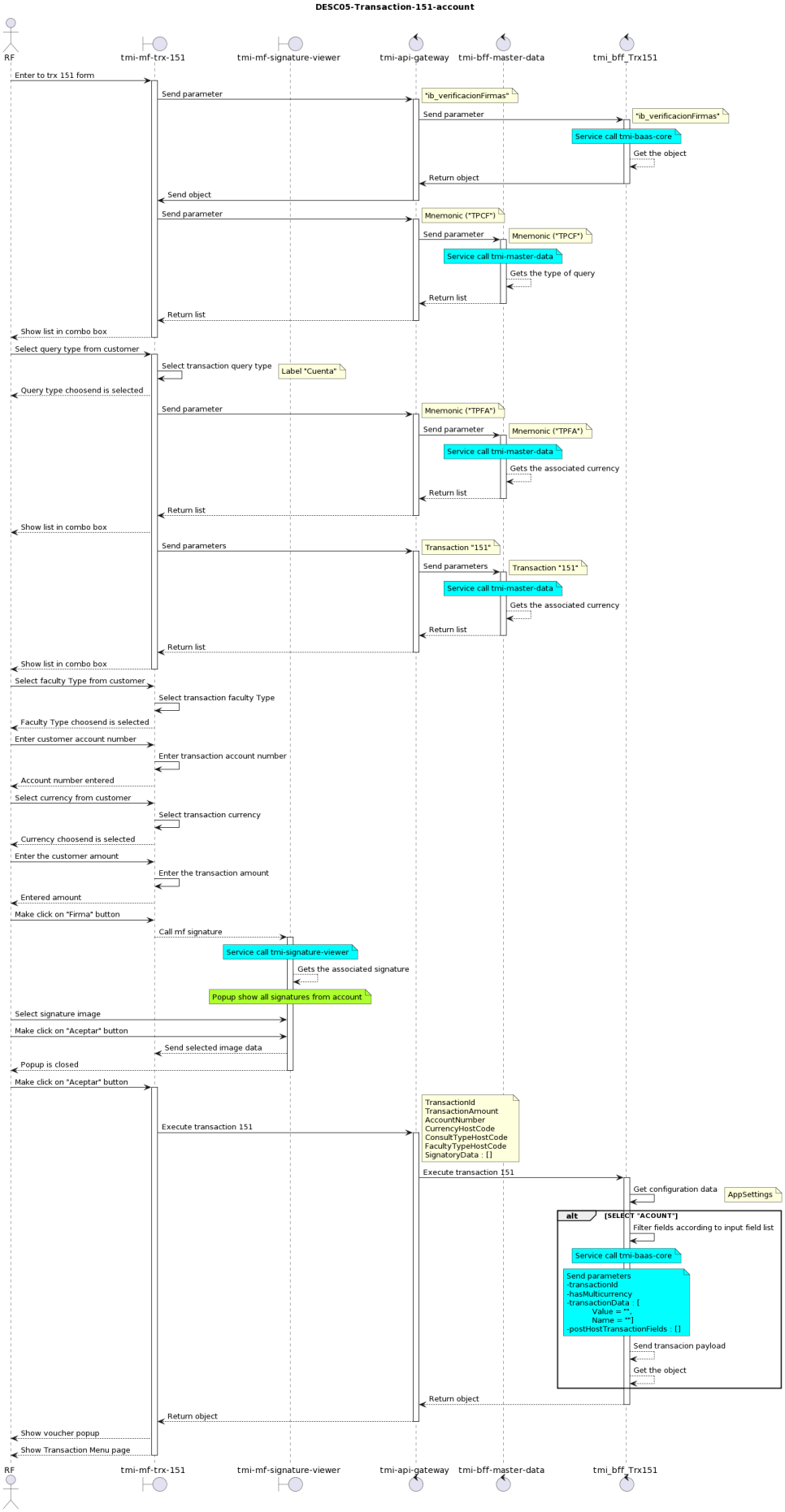
ANEXO N.° 9. Diseño UI de la arquitectura micro-frontend de "LA APLICACIÓN".



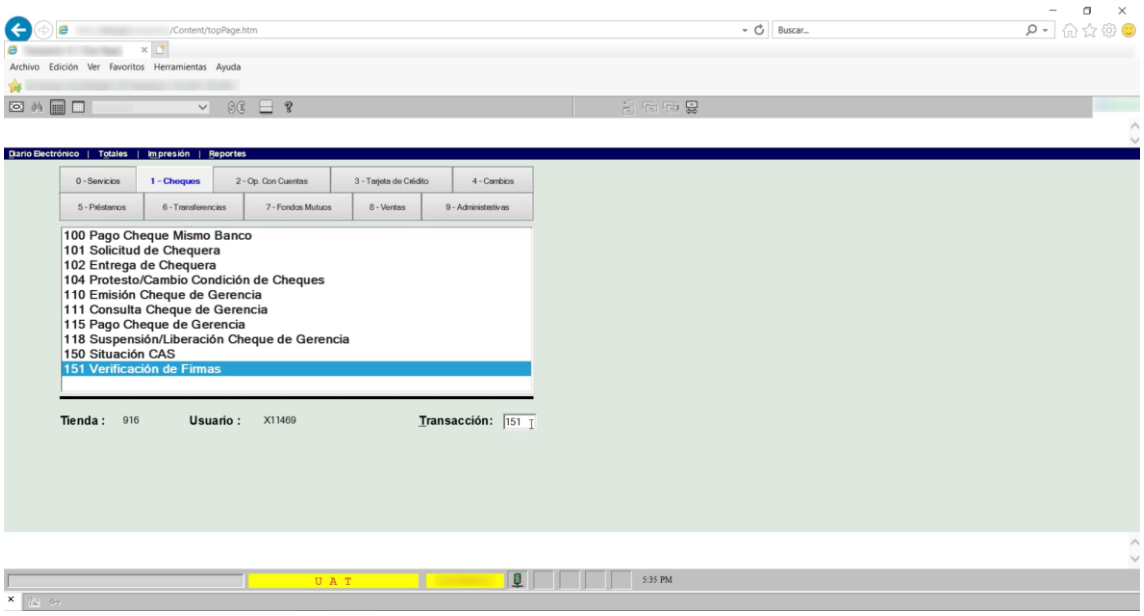


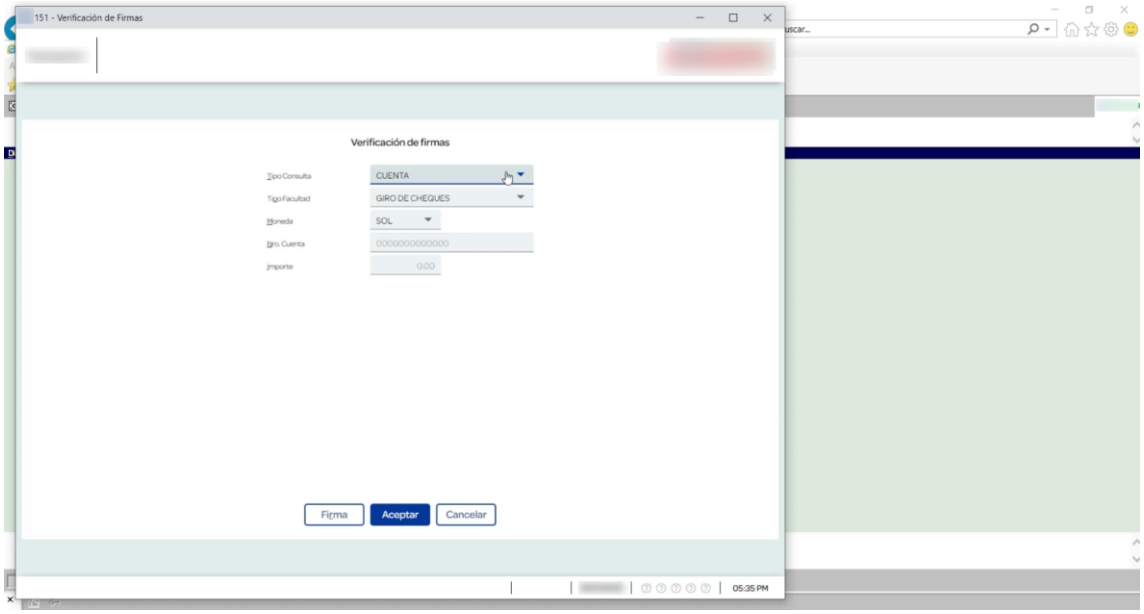
ANEXO N.° 10. Diagramas de secuencias transacción 151 de “LA APLICACIÓN”.

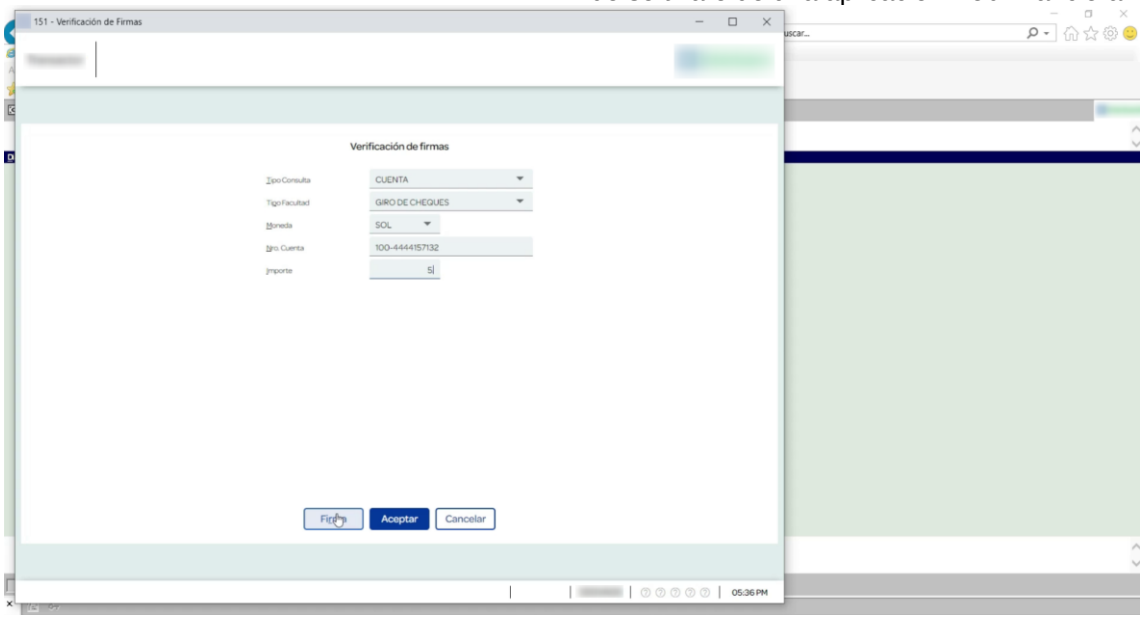


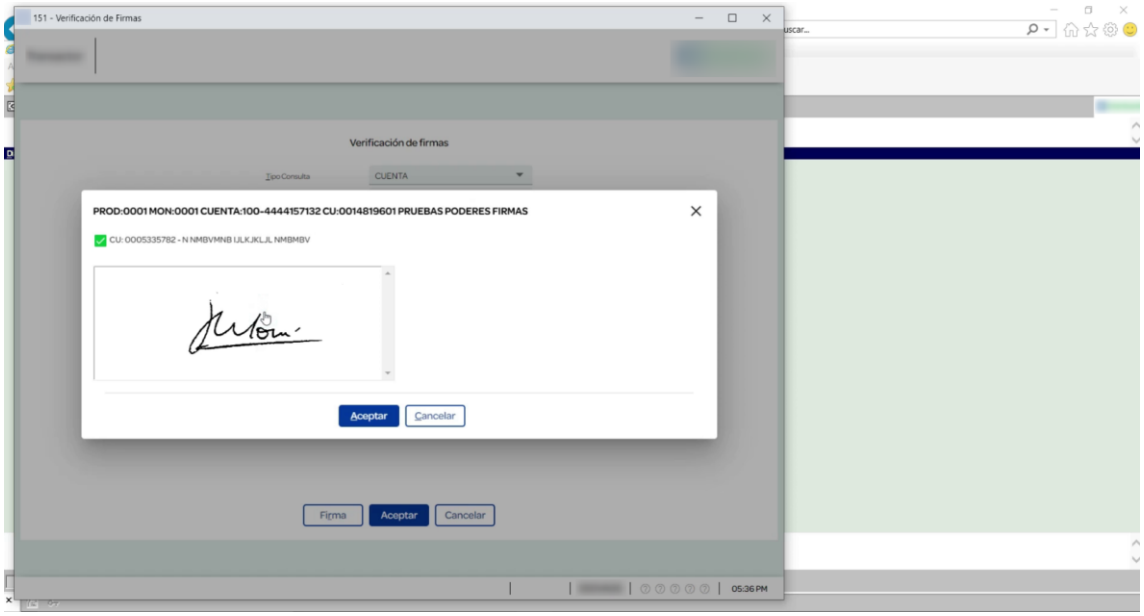

DESC05-Transaction-151-account
RF
tmi-mf-trx-151
tmi-mf-signature-viewer
tmi-api-gateway
tmi-bff-master-data
tmi_bff_Trx151
Enter to trx 151 form
Send parameter
"ib_verificacionFirmas"
Service call tmi-baas-core
Get the object
Return object
Send object
Mnemonic ("TPCF")
Service call tmi-master-data
Gets the type of query
Return list
Show list in combo box
Select query type from customer
Select transaction query type
Label "Cuenta"
Query type choosend is selected
Mnemonic ("TPFA")
Gets the associated currency
Send parameters
Transaction "151"
Select faculty Type from customer
Select transaction faculty Type
Faculty Type choosend is selected
Enter customer account number
Enter transaction account number
Account number entered
Select currency from customer
Select transaction currency
Currency choosend is selected
Enter the customer amount
Enter the transaction amount
Entered amount
Make click on "Firma" button
Call mf signature
Service call tmi-signature-viewer
Gets the associated signature
Popup show all signatures from account
Select signature image
Make click on "Aceptar" button
Send selected image data
Popup is closed
Make click on "Aceptar" button
TransactionId
TransactionAmount
AccountNumber
CurrencyHostCode
ConsultTypeHostCode
FacultyTypeHostCode
SignatoryData : []
Execute transaction 151
Get configuration data
AppSettings
alt
[SELECT "ACOUNT"]
Filter fields according to input field list
Send parameters
-transactionId
-hasMulticurrency
-transactionData : [
Value = ""
Name = ""]
-postHostTransactionFields : []
Send transacion payload
Get the object
Return object
Show voucher popup
Show Transaction Menu page

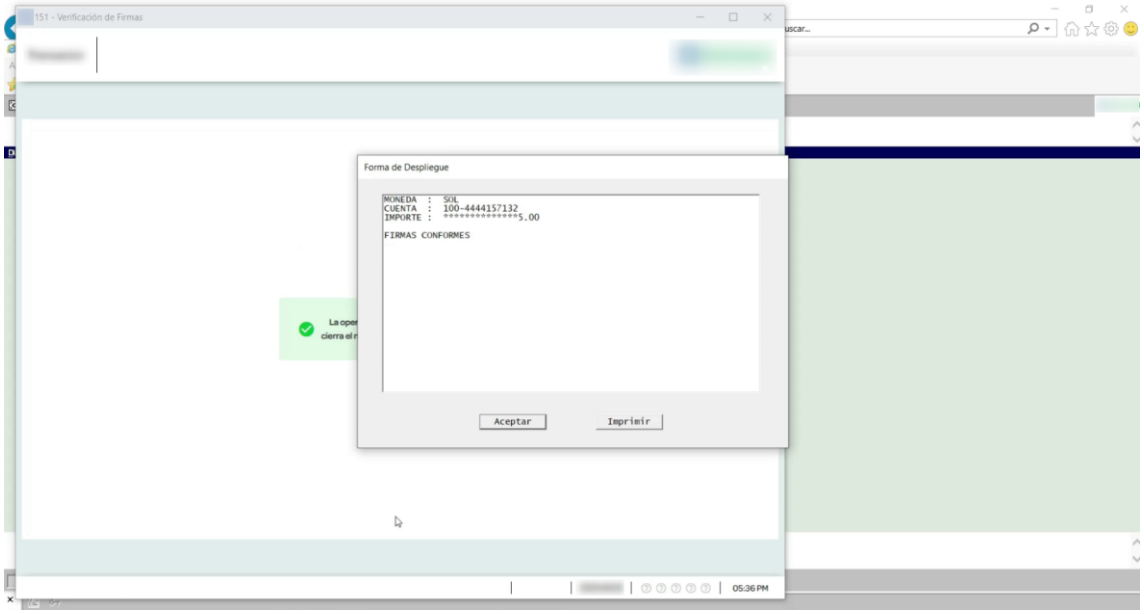
ANEXO N.° 11. Aplicación web con arquitectura micro-frontend de transacción 151 Verificación de firmas.

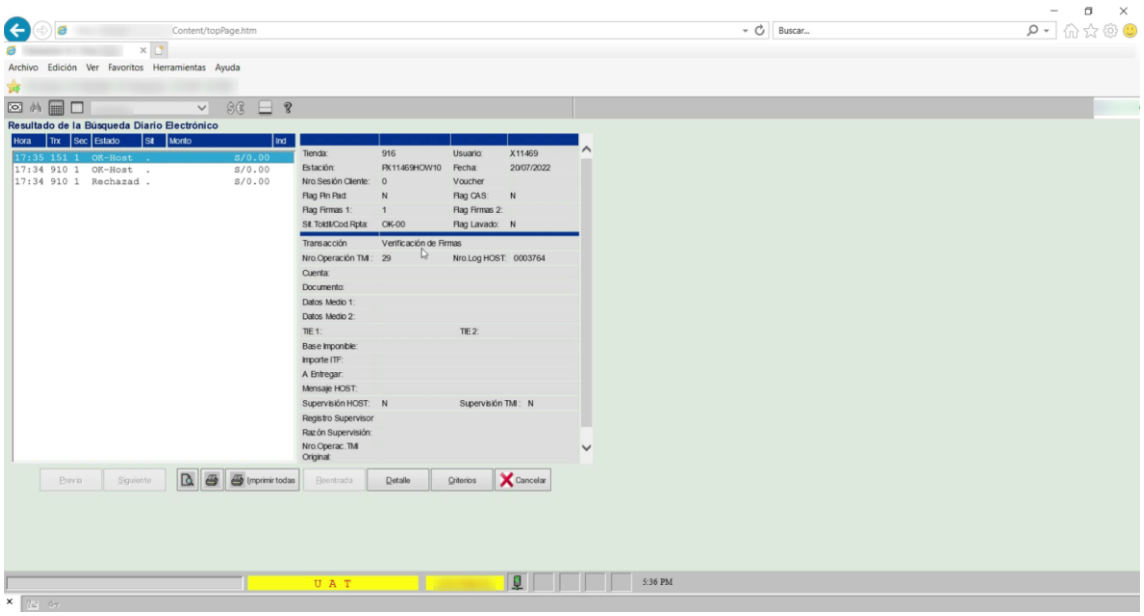





151 - Verificación de Firmas
Verificación de firmas
Tipo Consulta
CUENTA
Tipo Facultad
GIRO DE CHEQUES
Moneda
SOL
Nro. Cuenta
100-4444157132
Importe
5|
Firma
Aceptar
Cancelar
05:36 PM


151 - Verificación de Firmas
Verificación de firmas
Tipo Consulta
CUENTA
PROD:0001 MON:0001 CUENTA:100-4444157132 CU:0014819601 PRUEBAS PODERES FIRMAS
CU: 0005335782 - N NMBVMNB I,LKJKL,L NMBMBV
Aceptar
Cancelar
Firma
Aceptar
Cancelar
05:36 PM


151 - Verificación de Firmas
Forma de Despliegue
MONEDA : SOL
CUENTA : 100-4444157132
IMPORTE : ********5.00
FIRMAS CONFORMES
Aceptar
Imprimir
05:36 PM


Content/topPage.htm
Buscar...
Archivo Edición Ver Favoritos Herramientas Ayuda
Resultado de la Búsqueda Diario Electrónico
Hora Trx Sec Estado Sit Monto Ind
17:35 151 1 OK-Host . S/0.00
17:34 910 1 OK-Host . S/0.00
17:34 910 1 Rechazad . S/0.00
Tienda: 916 Usuario: X11469
Estación: PX11469HCW10 Fecha: 20/07/2022
Nro.Sesión Cliente: 0 Voucher
Flag Pin Pad: N Flag CAS: N
Flag Firmas 1: 1 Flag Firmas 2:
Sit.Tokit/Cod.Rpta: OK-00 Flag Lavado: N
Transacción Verificación de Firmas
Nro.Operación TMI: 29 Nro.Log HOST: 0003764
Cuenta:
Documento:
Datos Medio 1:
Datos Medio 2:
TIE 1: TIE 2:
Base Imponible:
Importe ITF:
A Entregar:
Mensaje HOST:
Supervisión HOST: N Supervisión TMI: N
Registro Supervisor
Razón Supervisión:
Nro.Operac. TMI Original:
Previa Siguiente Imprimir todas Reentrada Detalle Criterios Cancelar
U A T
5:36 PM

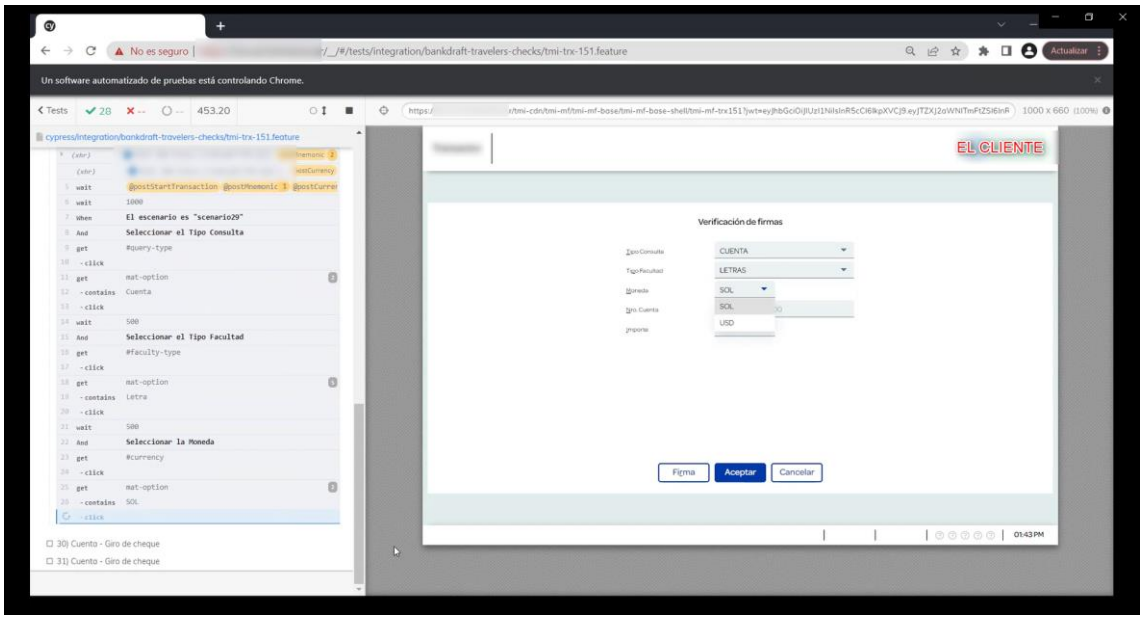
ANEXO N.° 12. Matrix de casos de pruenas de la transacción 151 Verificación de firmas.

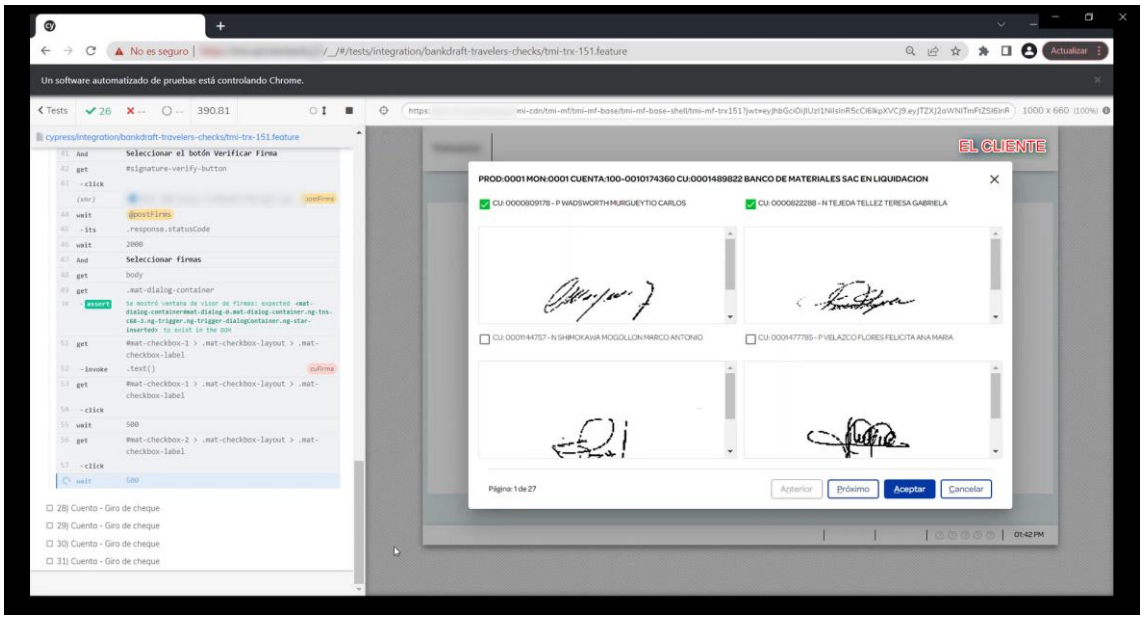
EL CLIENTE

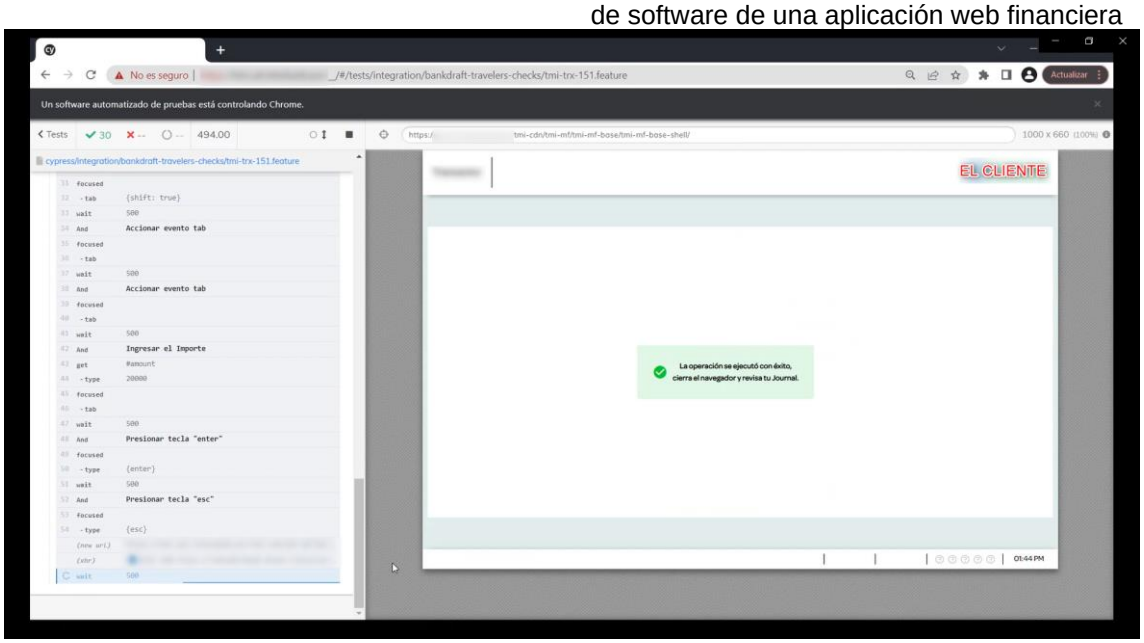
MATRIZ DE CASOS DE PRUEBA - TRX151 VERIFICACIÓN DE FIRMAS

						Detalle casos de prueba								
N. Cf Automa	Automati	N C	Ej t	Tipo prue	Aplica o	Flujo	Caso de Prueba	Pasos	Tip Consi	Val	Tipo Faculta	Nive	Tiem	Resultado esperado Actualizado	Tiem	Estat
CP-01	SI	1		Alcance		TRX151 Verificacion de firmas.	Usuario ingresa a la TRX 151 Verificacion de firmas. Cta con 1 firma(s) registrada(s), para el Tipo de Facultad: Giro de Cheques	1. Ingresar a la opción 1. Cheques 2. Seleccionar la opción 151 Verificación de Firmas 3. Seleccionar el Tipo Consulta: CUENTA 4. Seleccionar el Tipo Facultad: GIRO DE CHEQUE 5. Seleccionar la Moneda: SOL 6. Ingresar Nro. Cuenta (000444457032) 7. Ingresar el Importe 8. Seleccionar el botón Firma ..... 9. Seleccionar la firma ..... 10. Seleccionar el botón Aceptar ..... 11. Seleccionar el botón Aceptar	Cuenta	Cta con 1 firma(s) registrada(s)	Giro de Cheques	RF1	100	Se muestra la firma asociada a cuenta (Por cuenta) ..... Se habilita el botón Aceptar ..... Se cierra la ventana de firmas asociadas ..... Se muestra el Menú Principal con el mensaje: TRANSACCION OK y el Display con la MONEDA, CUENTA, IMPORTE y FIRMAS CONFORMES. Se muestra en JOURNAL el campo de FLAG FIRMAS 1. con valor 1		
CP-02	SI	2	SI	Alcance		TRX151 Verificacion de firmas.	Usuario ingresa a la TRX 151 Verificacion de firmas. Cta errada para el Tipo de Facultad: Transferencia y giro de cheque.	1. Ingresar a la opción 1. Cheques 2. Seleccionar la opción 151 Verificación de Firmas 3. Seleccionar el Tipo Consulta: CUENTA 4. Seleccionar el Tipo Facultad: TRANSFERENCIA Y GIRO DE CHEQUE 5. Seleccionar la Moneda: SOL 6. Ingresar Nro. Cuenta (000700009525) 7. Ingresar el Importe 8. Seleccionar el botón Firma	Cuenta	Cta errada	Transferencia y giro de cheque.	RF1	100	- Se muestra mensaje: "Número de cuenta errado"		
CP-03	SI	3		Alcance		TRX151 Verificacion de firmas.	Usuario ingresa a la TRX 151 Verificacion de firmas. Cta con 2 firma(s) registrada(s), para el Tipo de Facultad: Giro de Cheques	1. Ingresar a la opción 1. Cheques 2. Seleccionar la opción 151 Verificación de Firmas 3. Seleccionar el Tipo Consulta: CUENTA 4. Seleccionar el Tipo Facultad: GIRO DE CHEQUE 5. Seleccionar la Moneda: SOL 6. Ingresar Nro. Cuenta (000700000716) 7. Ingresar el Importe 8. Seleccionar el botón Firma ..... 9. Seleccionar firmas ..... 10. Seleccionar el botón Aceptar ..... 11. Seleccionar el botón Aceptar	Cuenta	Cta con 2 firma(s) registrada(s)	Giro de Cheques	RF1	100	Se muestra las firmas asociadas a cuenta (Por cuenta) ..... Se habilita el botón Aceptar ..... Se cierra la ventana de firmas asociadas ..... Se muestra el Menú Principal con el mensaje: TRANSACCION OK y el Display con la MONEDA, CUENTA, IMPORTE y FIRMAS CONFORMES. Se muestra en JOURNAL el campo de FLAG FIRMAS 1. con valor 2		
CP-04	SI	4	SI	Alcance		TRX151 Verificacion de firmas.	Usuario ingresa a la TRX 151 Verificacion de firmas. Campo Cta. Vacio para el Tipo de Facultad: Giro de Cheques	1. Ingresar a la opción 1. Cheques 2. Seleccionar la opción 151 Verificación de Firmas 3. Seleccionar el Tipo Consulta: CUENTA 4. Seleccionar el Tipo Facultad: GIRO DE CHEQUE 5. No ingresar Nro. Cuenta 6. Seleccionar el botón Firma	Cuenta	Campo Cta. Vacio	Giro de Cheques	RF1	100	- Se muestra mensaje: "El campo debe tener un valor para validar las firmas" - Si se selecciona el botón Aceptar se muestra el mensaje: "El campo debe tener un valor para validar las firmas" - También se pintan de color rojo los campo que faltan obligatorios y se coloca el foco en el campo Nro Cuenta.		
CP-05	SI	5		Alcance		TRX151 Verificacion de firmas.	Usuario ingresa a la TRX 151 Verificacion de firmas. Cta con 2 firma(s) registrada(s), para el Tipo de Facultad: Transferencia y giro de cheque.	1. Ingresar a la opción 1. Cheques 2. Seleccionar la opción 151 Verificación de Firmas 3. Seleccionar el Tipo Consulta: CUENTA 4. Seleccionar el Tipo Facultad: TRANSFERENCIA Y GIRO DE CHEQUE 5. Seleccionar la Moneda: SOL 6. Ingresar Nro. Cuenta (000700000716) 7. Ingresar el Importe 8. Seleccionar el botón Firma ..... 9. Seleccionar las 2 firmas ..... 10. Seleccionar el botón Aceptar ..... 11. Seleccionar el botón Aceptar	Cuenta	Cta con 2 firma(s) registrada(s)	Transferencia y giro de cheque.	RF1	274	Se muestra las firmas asociadas a cuenta (Por cuenta) ..... Se habilita el botón Aceptar ..... Se cierra la ventana de firmas asociadas ..... Se muestra el Menú Principal con el mensaje: TRANSACCION OK y el Display con la MONEDA, CUENTA, IMPORTE y FIRMAS CONFORMES. Se muestra en JOURNAL el campo de FLAG FIRMAS 1. con valor 2		

ANEXO N.° 13. Pruebas automatizadas de la transacción 151 Verificación de firmas.







ANEXO N.° 14. Correo de conformidad de pase OC 103246 a ambiente Productivo transacción 151 – Marcha Blanca.

RE: VERIFICACIÓN POST PASE - Orden de Cambio N° 103246 – PRD - TMI(M) - Despliegue de co...

CY

Reenviar

miércoles 22/06/2022 22:19

EVOL.Vargas Jacinto, Romario;

CC **Lista Soporte Cambios TI**; **Lista PEP**

conforme

De:
Enviado el: miércoles, 22 de junio de 2022 22:19
Para: EVOL.Vargas Jacinto, Romario
Asunto: RE: VERIFICACIÓN POST PASE - Orden de Cambio N° 103246 – PRD - TMI(M) - Despliegue de componentes para habilitar la TRX151 en nuevo front - Piloto - No genera indisponibilidad

Hola por favor tu conformidad funcional sobre la oc-103246 - Despliegue de componentes para habilitar la TRX151 en nuevo front

ANEXO N.° 15. Correo de conformidad de pase OC 106274 a ambiente Productivo transacción 151– Pase a nivel Nacional.