UNIVERSIDAD POLITÉCNICA SALESIANA ECUADOR

POSGRADOS

MAESTRÍA EN

SOFTWARE CON MENCIÓN EN DESARROLLO WEB Y MÓVIL

RPC-SO-34-NO.778-2021

OPCIÓN DE TITULACIÓN:

PROYECTO DE TITULACIÓN CON COMPONENTES DE INVESTIGACIÓN APLICADA Y/O DE DESARROLLO

TEMA:

CONSTRUCCIÓN DE UNA ARQUITECTURA DE CONTRASEÑAS DE UN SOLO USO OTP'S PARA TRANSACCIONES ELECTRÓNICAS DE PEQUEÑAS Y MEDIANAS INSTITUCIONES FINANCIERAS

AUTOR(ES)

ALEX FABIAN YAMBA AUQUI

DIRECTOR:

DANIEL GIOVANNY DÍAZ ORTIZ

QUITO – ECUADOR

2023

Autor(es):

Alex Fabian Yamba Auqui
Ingeniero de Sistemas
Candidato a Magíster en Software con Mención en Desarrollo Web y Móvil por la Universidad Politécnica Salesiana – Sede Quito.
ayamba@est.ups.edu.ec

Dirigido por:

Daniel Giovanny Díaz Ortiz
Ingeniero de Sistemas
Magíster en Redes de Comunicaciones
Master Universitario en Software Libre
ddiaz@ups.edu.ec

Alex Fabian Yamba Auqui
CONSTRUCCIÓN DE UNA ARQUITECTURA DE CONTRASEÑAS DE UN SOLO USO OTP'S PARA TRANSACCIONES ELECTRÓNICAS DE PEQUEÑAS Y MEDIANAS INSTITUCIONES FINANCIERAS

DEDICATORIA

Este trabajo es un reflejo no solo de mis esfuerzos, sino también de la generosidad y amor que he recibido de aquellos a mi alrededor.

A mi esposa, cuyo amor y apoyo han sido el refugio constante y la suave brisa en el intenso calor de los desafíos. Por las noches en vela y por el espacio en nuestras vidas que generosamente has compartido con mis estudios y mis pensamientos. Tu fortaleza y tu sacrificio son las alas invisibles que me han sostenido en este viaje.

A mis hijos, cuyo entusiasmo por la vida me recuerda todos los días la razón de mis esfuerzos. Perdonen las horas que no pasamos juntos; cada una de ellas ha sido invertida en construir un futuro en el que espero puedan florecer.

A mis padres, cuya sabiduría y amor incondicional son las raíces de donde he crecido. Vuestros sacrificios han sido el suelo fértil para mis propios logros.

A mi hermano, que, con su peculiar manera de involucrarse, siempre ha sabido cómo recordarme que la vida fuera de los libros también está llena de aprendizajes y momentos valiosos.

A la Cooperativa de Ahorro y Crédito Alianza de Valle, por su apoyo económico y moral que me ha permitido perseguir y alcanzar esta meta.

Cada página de esta tesis lleva la huella de todos ustedes, y es con un corazón lleno de gratitud que dedico este logro a cada uno de ustedes. Gracias por caminar conmigo en este viaje.

AGRADECIMIENTO

Mi primer y más profundo agradecimiento es para Dios, por haber iluminado mi camino, dado fortaleza en los momentos de desafío y ser la constante fuente de mi fe y esperanza. Estoy seguro, que su presencia ha sido decisiva en cada paso que he dado y en cada logro que he alcanzado.

Mi gratitud se extiende al Ing. Daniel Diaz cuya orientación ha sido fundamental en la realización de esta tesis. Su apoyo inquebrantable y su sabiduría han sido una luz guía en el proceso de investigación y redacción.

No puedo dejar de reconocer a la Cooperativa de Ahorro y Crédito Alianza de Valle por su generoso apoyo económico. Sin la confianza del Ing. Rolando Cadena, Ing. Elizabeth Orbea y el Ing. Daniel Mosquera, este paso en mi carrera no hubiera sido posible.

A mis compañeros de maestría, juntos hemos compartido retos y éxitos que han enriquecido mi experiencia dejando huella en mi vida académica.

A mi querida familia: mi esposa y mis hijos, por su amor incondicional, paciencia y comprensión. Su apoyo ha sido el pilar sobre el que he podido apoyarme en los momentos más difíciles.

Mis padres han sido fuente de inspiración constante. Su amor ha sido mi guía desde el principio, y las lecciones que he aprendido de ellos son la base sobre la cual he construido cada logro.

A los profesores y al personal de la Universidad Politécnica Salesiana, por su guía y apoyo en esta trayectoria del posgrado. Su dedicación ha hecho posible no solo la culminación de este trabajo, sino mi crecimiento personal y profesional.

Cada paso de este viaje ha estado acompañado por bendiciones y lecciones, y por ello, mi corazón rebosa gratitud.

Tabla de Contenido

Índice de Figuras

Índice de Tablas

Construcción de una arquitectura de contraseñas de un solo uso OTP's para transacciones electrónicas de pequeñas y medianas instituciones financieras

Autor(es):

ALEX FABIAN YAMBA AUQUI

Resumen

En el actual panorama digital, la seguridad en las transacciones en línea es de alta prioridad para las instituciones financieras, no únicamente para las grandes instituciones, sino también para las pequeñas y medianas, las cuales a menudo enfrentan restricciones de presupuesto y conocimientos especializados en desarrollo de proyectos seguros con enfoque en ciberseguridad. Esta tesis aborda esta problemática desarrollando una arquitectura para la generación de códigos de autenticación de un solo uso (OTP), enfocada en mejorar la seguridad en las transacciones monetarias de los clientes.

El objetivo principal de este proyecto es desarrollar una arquitectura que genere códigos OTP seguros y que sea accesible para un amplio espectro de instituciones financieras. Se realiza un análisis del estado actual de la tecnología OTP, examinando sus aplicaciones en entornos conectados (online) y desconectados (offline), y sus efectos en la seguridad transaccional. Además, se estudia el impacto de estas tecnologías en la prevención de fraudes y accesos no autorizados.

En el desarrollo de esta arquitectura, se ha diseñado y ejecutado una aplicación móvil, utilizando una metodología mixta que permite su adaptación e integración en diversas plataformas tecnológicas. Así mismo, se ha creado una API flexible y adaptable, la cual puede integrarse fácilmente en proyectos existentes o nuevos, fomentando así una mayor adopción de la solución propuesta.

Este trabajo resalta la necesidad de soluciones de seguridad accesibles y eficientes, no solo para instituciones con recursos económicos, sino también para aquellas pequeñas y medianas, democratizando la seguridad en el ecosistema financiero digital. El proyecto contribuirá significativamente al campo de la ciberseguridad financiera, ofreciendo una solución que equilibra costo y eficacia, y que puede ser implementada en una variedad de contextos institucionales, marcando así un paso adelante en la protección de transacciones en el ámbito digital.

Abstract

In the current digital landscape, online transaction security is of high priority for financial institutions, not only for large entities but also for small and medium-sized ones, which often face budget constraints and lack of specialized knowledge in developing secure projects with a focus on cybersecurity. This thesis addresses this issue by developing an architecture for the generation of one-time authentication codes (OTP), aimed at enhancing the security of client monetary transactions.

The main objective of this project is to develop an architecture that generates secure OTP codes and is accessible to a wide range of financial institutions. An analysis of the current state of OTP technology is conducted, examining its applications in connected (online) and disconnected (offline) environments, and its effects on transactional security. Furthermore, the impact of these technologies in preventing fraud and unauthorized access is studied.

In the development of this architecture, a mobile application has been designed and implemented, using a mixed methodology that allows its adaptation and integration into various technological platforms. Similarly, a flexible and adaptable API has been created, which can be easily integrated into existing or new projects, thus promoting a wider adoption of the proposed solution.

This work highlights the need for accessible and efficient security solutions, not only for institutions with economic resources but also for small and medium-sized ones, democratizing security in the digital financial ecosystem. The project will significantly contribute to the field of financial cybersecurity, offering a solution that balances cost and effectiveness, and can be implemented in a variety of institutional contexts, thus marking a step forward in the protection of transactions in the digital realm.

Palabras clave:

OTP (One-Time-Password), autenticación de dos factores, códigos de acceso de un solo uso, doble factor de autenticación, segundo factor de autenticación, generación de OTP offline, contraseñas temporales, token temporizado, token offline, transacciones financieras seguras.

1. Introducción

1.1 Antecedentes

Toda institución que ofrecen servicios en el ámbito financiero cuenta con métodos de validación de clientes, puede ser un método manual en el que se involucran personas o un método automático en el que se involucra un sistema de software, la transformación digital creciente, permite que los clientes financieros puedan hacer uso de plataformas informáticas para realizar transacciones con sus cuentas bancarias.

Las transacciones bancarias implican el transporte información de los clientes, los datos que viajan en la red podrían verse comprometidos si una aplicación web o una App no toma las acciones mínimas necesarias para asegurar que la información no sea alterada, plagiada o incidentes más críticos como comprometer las claves y usuarios de los clientes. Uno de los factores más robustos es utilizar un código OTP, se genera una única vez, tiene segundos de vida útil y no se repite en el transcurso del tiempo.

Este método lo utilizan gran parte de instituciones financieras grandes, sin embargo, las pequeñas y medianas aún tienen trabajo por hacer alrededor de este tema. El uso de la Apps en los dispositivos móviles facilita en gran medida interactuar con este tipo de autenticación para los usuarios, por ese motivo la mejor opción por facilidad y seguridad, es el uso de una App dedicada a generar OTP, por lo tanto, para este propósito se requiere una arquitectura segura y fiable que pueda brindar confianza a los clientes.

En 2022 el trabajo realizado por Krishna, Sandhya P., Tejasri, Duggempudi, Soumya, Bellamkonda, Madhuri, Maddi, Lubna, proponen la creación de una App Android para generar una OTP basada en tiempo, asegurando que la transmisión de datos se realice con algoritmos criptográficos de cifrado para que un atacante no pueda conocer la información que viaja.

La generación del OTP se basa en datos personales del usuario como nombre de usuario, número celular y una marca de tiempo. Para generar una OTP, el usuario ingresa su nombre de usuario y número de teléfono, se genera una clave a partir del número, que se rastrea y cifra con una OTP utilizando el algoritmo DES. Luego, se genera un carácter aleatorio como OTP utilizando el código encriptado y aleatorio.

Este trabajo cuenta con pruebas y estadísticas en donde se demuestra el trabajo de los algoritmos criptográficos DES y AES. Además, que, bajo el contexto de problemas conocidos de phishing, claves comunes utilizados por los usuarios, es necesario el uso de un OTP para tener más seguridad en las transacciones.

En 2021 el trabajo realizado por Naik, Rasika Singh, Udayprakash, propone crear un sistema que genere OTP de 6 dígitos con un tiempo de validez limitado, haciendo que sea extremadamente difícil para una computadora general descifrar la OTP con fuerza bruta.

A diferencia de los métodos tradicionales para autenticar, se propone generar un OTP basado en un mapa caótico, de tal manera que la aleatoriedad del código sea mucho más difícil de adivinar para un atacante, y que no pueda ser predecible.

Plantea la generación de OTP utilizando un mapa caótico B-exponencial. Según el concepto, cuando un usuario ingresa el nombre de usuario y la contraseña, se verifican las credenciales; si las credenciales son incorrectas, se solicita al usuario que ingrese el nombre de usuario y la contraseña nuevamente. Si las credenciales son correctas, se invoca el algoritmo de generación de OTP del lado del servidor. Una vez que se genera la OTP, se solicita al usuario que la ingrese y se valida la OTP. El sistema OTP permite que el usuario inicie sesión si ingresa una OTP válida. Si la OTP ingresada es incorrecta, se le solicita al usuario que ingrese el nombre de usuario y la contraseña nuevamente. Las entradas definidas por el usuario se pasan a través de tres mapas caóticos exponenciales B durante el proceso de generación de OTP. La salida de bucle cerrado del bucle de mapa caótico exponencial B se pondera y luego se agrega a la salida sin bucle. Esta suma es operada por módulo,

y el bit generado se compara con la entrada del usuario y la salida del mapa caótico B-exponencial. La evaluación final es un bit aleatorio y el proceso se repite 24 veces para producir la OTP final.

En 2016, el trabajo realizado por Kaur, Navpreet and Devgan, Mandeep y Bhushan, Shashi, propone una autenticación más sólida en los sistemas en línea. Implica el intercambio de semillas, un token basado en software a través del túnel Transport Layer Security (TLS) que se utiliza para generar contraseñas de un solo uso en línea.

El objetivo principal de este esquema es proporcionar una autenticación más sólida en los sistemas en línea. Implica el intercambio de semillas, un token basado en software a través del túnel Transport Layer Security (TLS) que se utiliza para generar contraseñas de un solo uso en línea.

Una aplicación móvil basada en Android que genera contraseñas de un solo uso a partir de una semilla secreta que se genera en el lado del servidor y se envía al lado del cliente. La contraseña de un solo uso se genera a partir de este valor semilla en ambos extremos, es decir, el extremo del cliente y el extremo del servidor, luego la autenticación solo ocurre si los códigos generados en ambos lados son los mismos.

Todo lo que viaja a lo largo de la red está encriptado usando encriptación AES de clave de 128 bits. Solo hay dos entidades que están involucradas en este mecanismo de autenticación.

La autenticación robusta de inicio de sesión está diseñada para mediante un método de autenticación de múltiples factores más seguro para los usuarios de Internet que utilizan algoritmos criptográficos y protocolos web existentes. A diferencia de los esquemas de autenticación anteriores, el valor semilla nunca viaja en forma de texto sin formato. Una característica de seguridad más que se agrega a este enfoque es que la OTP es generada por el dispositivo móvil en el que se ejecuta la aplicación.

1.2 JUSTIFICACIÓN

En los últimos años la transformación digital a nivel mundial ha obligado a comercializar todo tipo de productos mediante diferentes plataformas tecnológicas. Como no podía ser de otra manera, el medio financiero no es la excepción. La necesidad de poder realizar cualquier tipo de transacción financiera remotamente ha abierto un abanico de posibilidades. Entre estas, una de las más utilizadas es a través del uso de dispositivos móviles.

Las facilidades para el usuario crecen mediante el uso de apps y páginas web que ofrecen servicios específicos para satisfacer necesidades concretas. Así, también se incrementa la ciberdelincuencia, por lo tanto, para los creadores de sistemas se presentan retos cada vez más complejos de resolver. Uno de estos retos es ¿cómo validar realmente que un usuario que utiliza un canal digital es efectivamente la persona autorizada? En este sentido, se han creado varios métodos de identificación que validan la autenticidad de un usuario.

A diario en nuestro entorno, se presentan denuncias judiciales en las cuales un usuario de una plataforma digital indica haber sido hackeado. Dentro de esto, se señala el robo de dinero en forma de transferencia, misma que el usuario niega haber realizado. Los ataques elaborados por la ciberdelincuencia varían entre ataques de fuerza bruta, phishing e ingeniería social. En estas dos últimas, es el mismo usuario el que entrega información que compromete y, en varias ocasiones, permite vulnerar y ejecutar las transacciones sin autorización conocida por el verdadero propietario de una cuenta; es decir, en este punto un atacante ha logrado suplantar la identidad de un usuario.

El Factor de Doble Autenticación se destaca como uno de los métodos más empleados y eficaces en la verificación de la autenticidad de un usuario. Su popularidad y seguridad, sin embargo, están condicionadas por la calidad de su implementación. Este método no solo fortalece la seguridad percibida por el usuario final, sino que también asegura un alto nivel de servicio proporcionado por las empresas a través de sus aplicaciones o sitios web transaccionales.

Actualmente, uno de los medios más utilizados para realizar transacciones electrónicas es el One Time Password, más conocido como OTP. Para el envío y recepción de un OTP se lo realiza mediante SMS, implicando un costo por el uso de la red de las operadoras telefónicas. Así también, como se había mencionado, si bien es cierto es uno de los medios más seguros, sigue siendo susceptible a vulnerabilidades que pueden comprometer la integridad de un proceso.

Cuando se trata de términos financieros, es sumamente importante para el cliente mantener sus cuentas seguras, así como para las instituciones financieras ofrecer un medio seguro que garantice que sus transacciones en canales digitales sean únicamente autorizadas por el propietario de la cuenta bancaria. Aquí es donde el OTP toma un rol muy valioso para las entidades financieras, implementando un mecanismo robusto de autenticación, conocido única y exclusivamente por el usuario cliente.

Un OTP es un código único generado a partir de datos personales del usuario. Otra forma de generar estos códigos implica el uso de semillas y marcas de tiempo, asegurando un código numérico de 6 dígitos que no podrá ser utilizado más de una vez y que no se repetirá entre usuarios. Este código, totalmente aleatorio, tiene un tiempo de expiración: después de 30 segundos, el código queda caducado e inutilizable, generándose inmediatamente un nuevo código. Para generar los OTP, el usuario debe utilizar una aplicación específica para esta función. Antes de generar los OTP, el usuario debe registrarse validando información personal, como nombres, correo electrónico, número de teléfono y usuario de la cuenta digital.

Las pequeñas y medianas empresas (PYMES) del sector financiero en nuestro país a menudo se encuentran aisladas de estos temas de seguridad, ya sea por desconocimiento, limitaciones presupuestarias o por la cantidad de transacciones que generan. Estas empresas no suelen contar con este tipo de métodos de seguridad, que podrían contribuir significativamente al crecimiento digital de su organización. Por esta razón, se propone apoyar a las PYMES para que puedan ofrecer servicios más seguros, variados y de mejor calidad a través de los diferentes canales digitales.

1.3 Objetivos

2.1.1 Objetivo General

- Desarrollar una arquitectura para la generación de códigos OTP, para asegurar las transacciones monetarias de un cliente.

2.1.2 Objetivos Específicos:

- Analizar el estado de arte de OTP y cuáles son sus implicaciones en su aplicación de manera offline y online.

- Diseñar una arquitectura segura de generación de OPT´s que garanticen su unicidad, confidencialidad, e integridad.

- Diseñar una aplicación móvil capaz de acoplar la arquitectura de generación de OPT´s utilizando una metodología mixta.

- Realizar pruebas de funcionamiento que permitan evaluar el adecuado desempeño de la arquitectura.

- Desarrollar la API correspondiente a partir del diseño y evaluación para que pueda ser integrada en cualquier proyecto.

1.4 Alcance

El estado del arte dentro de un proyecto constituye la base fundamental para abordar el tema propuesto. En este sentido, se revisarán al menos diez investigaciones sobre OTPs, utilizando fuentes bibliográficas reconocidas como Scopus, IEEE Xplore, Scielo, entre otras. Además, se consultará información disponible en diversas plataformas en línea, las cuales constituyen una fuente importante de datos y perspectivas relevantes para el avance del proyecto.

El diseño de una arquitectura segura para la generación de OTPs se logrará mediante la implementación de patrones de desarrollo estándares, la aplicación de principios SOLID, código limpio y diversos estándares vigentes en el ámbito del software. Parte de la arquitectura incluye el enrolamiento del usuario en la plataforma, con una verificación automática de la información. Se emplearán algoritmos robustos de encriptación para asegurar que los datos no viajen en texto plano. A nivel técnico, la arquitectura se basará en tecnologías de Microsoft y Google, utilizando IDEs y frameworks actuales.

Crear una arquitectura para una aplicación móvil destinada a generar y desplegar códigos OTP, que sirvan como segundo factor de autenticación. Esto permitirá que los usuarios realicen procesos transaccionales en una página web, tales como inicio de sesión, transferencias, cambios de clave y otras transacciones que, por razones de seguridad, requieran una capa adicional de validación de la identidad del usuario.

Diseñar una aplicación móvil que sea fácil de utilizar para el usuario, la cual incluirá la funcionalidad de generar códigos OTP. Esta app permitirá generar códigos un número ilimitado de veces, mostrando al cliente el tiempo restante antes de la caducidad del código y regenerándolo automáticamente. Cada código será único y exclusivo para cada usuario, asegurando que no se puedan compartir entre distintos usuarios.

Una parte importante de cualquier proyecto de software son las pruebas, las cuales garantizan la fiabilidad y confianza en el uso de una aplicación. En este caso, se

realizarán pruebas directamente dentro de la app, que, al ser de arquitectura offline, no debería enfrentar problemas de sobrecarga. Las pruebas incluirán escenarios totalmente desconectados de internet, sin señal celular, en modo avión, así como también con conexión a internet. Además, se validará que un usuario no pueda registrarse más de una vez. Se verificará la caducidad del OTP y se llevarán a cabo pruebas para garantizar que los OTP generados no se repitan, incluso ante múltiples peticiones.

Desarrollar una arquitectura de APIs que permita la integración con las diferentes instituciones financieras. Las APIs serán creadas siguiendo formatos estándares de comunicación y transporte de información. Para ello, se utilizarán protocolos HTTPS y formatos JSON en el transporte de información, incorporando también cifrado para los datos sensibles de los usuarios.

Debido a la naturaleza de la arquitectura, que permitirá generar los OTPs dentro de la misma aplicación y de manera offline, se busca que las pequeñas y medianas empresas (PYMES) dejen de depender de un servicio de SMS. Así, el código se mostrará de manera inmediata en la pantalla del dispositivo móvil, facilitando un acceso rápido y eficiente para el usuario.

Una vez que el OTP se haya generado y mostrado en la aplicación, este tendrá un tiempo de vida útil limitado y no podrá ser utilizado nuevamente. Además, el OTP no se podrá usar más de una vez. La aplicación también incluirá una pre-validación de los datos del usuario, lo que implica el registro con una identificación y contraseña creadas previamente para poder registrar la aplicación en nuestro dispositivo móvil personal. Esta no podrá ser utilizada en otro dispositivo.

El proyecto no incluye la integración con ninguna institución financiera. Sin embargo, se tiene previsto un demo que demuestre el funcionamiento del OTP. Esto consistirá en una página web de ejemplo, diseñada para validar el código generado y mostrado en la aplicación.

1.5 Metodología

Para abordar los temas de seguridad en el área de informática, resulta esencial investigar en diversos tipos de fuentes. Esto permite obtener un panorama más claro sobre lo que se desea realizar. En los últimos años, ha sido de vital importancia asegurar las transacciones financieras en las plataformas tecnológicas a las que los clientes tienen acceso.

Crear una arquitectura funcional y segura para la generación de OTPs representa un gran aporte para las instituciones financieras pequeñas y medianas. Con el objetivo de lograr un aporte de calidad, se han realizado investigaciones a partir de diversas fuentes, incluyendo internet y bases de datos bibliográficas. Además, se han revisado conceptos de algoritmos criptográficos para asegurar el transporte seguro de la información en la red.

Debido a que el dispositivo móvil se ha convertido en un aliado esencial para acceder a todo tipo de contenido en la red, se desarrollará una arquitectura específica para la generación de OTPs mediante el celular.

Para obtener información que aporte ideas y facilite el entendimiento del funcionamiento de los OTP, se han llevado a cabo investigaciones cualitativas a través de búsquedas en bases de datos bibliográficas especializadas, como IEEE Xplore, Scopus y Web of Science, entre otras. Respecto al desarrollo de aplicaciones móviles, se ha investigado una variedad de tecnologías, incluyendo Android, React Native y Flutter. Tras realizar pruebas de concepto, se ha determinado que Flutter es la herramienta más adecuada para apoyar en el cumplimiento de los objetivos.

En términos técnicos, se han revisado tres de las metodologías ágiles más conocidas en el mercado: SCRUM, XP y Kanban. Basándonos en la experiencia y el conocimiento adquiridos sobre estas metodologías, se optará por una metodología mixta para el desarrollo del proyecto. Esta aproximación integrará las mejores prácticas de las tres metodologías analizadas, y se enfocará en utilizar periodos

definidos de tiempo para generar entregables, recibir retroalimentación y avanzar a través de varias fases en el desarrollo del código.

2. Marco teórico referencial

2.1 Aplicaciones existentes

Con la creciente ola de ataques generados a nivel mundial, no es de extrañar que los gigantes tecnológicos hayan tomado cartas en el asunto y por ende disponibilicen aplicaciones de tokens de seguridad bajo sus propias marcas. Microsoft y Google cuentan con sus aplicaciones de tokens de seguridad para apoyar a los usuarios en el aseguramiento de sus cuentas de correo entre otros servicios digitales.

2.1.1 Revisión global

Microsoft Authenticator y Google Authenticator son las aplicaciones conocidas a nivel global, estás permiten la generación de Otps temporizados, cada una con sus características particulares son una muy buena alternativa a la hora de proteger las cuentas de sus usuarios.

2.1.2 Revisión local

A nivel local y con enfoque en las instituciones financieras, únicamente las grandes han optado por la utilización de una aplicación de este estilo para mitigar los posibles ataques como suplantación de identidad, los ataques de hombre en el medio, el phishing, entre otros. A continuación, se presenta algunas de las instituciones financieras del medio que hacen uso de este mecanismo como segundo factor de autenticación Tabla 1:

Tabla 1.
Aplicaciones Utilizadas Localmente

Institución	App	Descripción	Fuente
Banco Pacifico	PacificId	Con esta app móvil Genera códigos OTP para poder realizar pagos y cobros de forma segura.	https://www.bancodelpacifico.com/personas/servicios/servicios-generales/pacificid
Produbanco	Entrust Identity	Es una aplicación para la autenticación y verificación de transacciones mediante un OTP.	https://www.entrust.com/es/solutions/user-identity
Jep	JepToken	Es una aplicación móvil que permite generar OTP para realizar transacciones en línea de forma rápida.	https://www.jep.coop/productos-servicios/en-linea/jeptoken

2.2 Normativas y estándares de seguridad

En el ámbito de las transacciones financieras digitales, ya sean monetarias o no monetarias, las instituciones reguladoras establecen una serie de parámetros a seguir para garantizar un servicio excelente al usuario, enfocándose en la seguridad de sus datos y transacciones. Esto aplica tanto para las interacciones a través de una página web como para el uso de una aplicación en dispositivos móviles.

2.2.1 Medidas de seguridad tecnológicas para transferencias electrónicas

La forma en que se realizan las transacciones financieras a nivel mundial ha evolucionado de manera inimaginable hasta hace unos años. Sin embargo, de manera proporcional, también ha aumentado el riesgo de amenazas a la seguridad en los canales digitales, tanto web como móviles. El fraude se ha convertido en uno de los factores críticos de riesgo para los usuarios de estos canales.

Particularmente, las empresas financieras medianas y pequeñas, y especialmente estas últimas, son más vulnerables a riesgos y brechas de seguridad. Esto se debe a que comúnmente carecen de la tecnología y recursos necesarios para garantizar la seguridad en las transacciones digitales. Por esta razón, la Superintendencia de Economía Popular y Solidaria incluye, dentro de sus resoluciones, una sección específica dedicada a asegurar el uso seguro de las transferencias digitales.

2.2.2 Resolución No. SEPS-IGT-IR-ISF-ITIC-IGJ-2017-103

A continuación, se describen de manera general varios de los puntos considerados en la resolución de la SEPS.

a. **Encriptación de información en plataformas digitales:** Utilizar métodos verificados y probados que permitan ofuscar la información que se transmite a través de la red. Todas las plataformas digitales deben contar con procesos de encriptación para evitar comprometer la confidencialidad de la información de los usuarios.
b. **Precautelar la integridad y privacidad de registros de datos de usuarios:** Es fundamental implementar medidas y prácticas de seguridad como cifrado de los datos, gestión de contraseñas, registros y auditoria, seguridad de la red, actualizaciones y parches de seguridad, capacitaciones a los usuarios, políticas y procedimientos de seguridad, evaluación y monitoreo de riesgos, copias de seguridad y recuperación entre otros.

c. **Límites para las transferencias enviadas:** Tanto la institución como el usuario deberán tener la posibilidad de parametrizar los montos límites que se van a permitir transaccionar mediante transferencias enviadas. En el caso del usuario, es recomendable que para autorizar o modificar los montos de transacción, se realice mediante un segundo factor de autenticación como es OTP.
d. **Consultas, Validaciones, Acreditación y Débitos:** Todas las acciones sobre el saldo del cliente deben realizarse en tiempo real, de tal forma que el cliente pueda saber de manera inmediata cualquier afectación que ha recibido su saldo, además de contar con un sistema de notificaciones inmediatas hacia el cliente ya sea vía email y/o SMS.
e. **Contar con medidas de autorización y autenticación:** El medio de financiero para mitigar riesgos de fraude como suplantación de identidad, debe contar con al menos dos de los tres factores estándares en el proceso de autenticación, "algo que se sabe, algo que se tiene, o algo que es", tomando en cuenta que uno de estos debe ser dinámico cada vez que se realiza una transacción y otra debe ser una clave de un solo uso como OTP(One Time Password).

2.2.3 PCI DSS

PCI DSS (Payment Card Industry Data Security Standard) es un conjunto de estándares de seguridad para la protección de datos de tarjetas de pago. Fue creado por las principales marcas de tarjetas de crédito, como Visa, Mastercard, American Express, JCB y Discover, con el objetivo de garantizar la seguridad y protección de los datos confidenciales de las tarjetas de pago.

El PCI DSS establece un conjunto de requisitos de seguridad que deben ser cumplidos por cualquier organización que procese, transmita o almacene datos de tarjetas de pago. Estos requisitos incluyen medidas de seguridad físicas y lógicas,

controles de acceso, políticas de gestión de contraseñas, cifrado de datos, pruebas de seguridad regulares y auditorías externas.

El cumplimiento del PCI DSS es obligatorio para todas las organizaciones que procesan, transmiten o almacenan datos de tarjetas de pago, independientemente de su tamaño o ubicación geográfica. El no cumplimiento de los requisitos del PCI DSS puede resultar en multas y sanciones financieras significativas, así como en la pérdida de la confianza y reputación del cliente.

3. Desarrollo del proyecto

3.1 Introducción

El manejo y administración de dinero, forma parte natural de casi todas las actividades que realizan las personas en su diario vivir, sin ese activo que se utiliza para todos los intercambios, el ser humano no podría realizar ninguna operación que conlleve movimientos monetarios. Con la increíble velocidad que crece el desarrollo de plataformas tecnológicas para realizar cualquier tipo de transacción al alcance de nuestro computador o celular, todos los procesos bancarios han volcado sus esfuerzos en ser automatizados, dando como resultado miles de transacciones bancarias sin límites regionales facilitando el envío y recepción de dinero de un país a otro en cuestión de minutos.

Como parte fundamental de toda transacción monetaria está contemplada la seguridad para garantizar que el dinero llegue desde su portador inicial hasta su nuevo beneficiario sin que sea interceptado y llevado por agentes externos no autorizados a tomar ese dinero.

Uno de los métodos más utilizados para realizar cualquier transacción monetaria mediante un servicio digital bancario es con la utilización de usuario y contraseña, esto garantiza de cierta manera que únicamente el propietario de una cuenta pueda hacer uso de su dinero. En contraparte a los beneficios que tenemos con tantos servicios digitales bancarios, ha crecido a la par el uso de métodos para intervenir y hurtar dinero de cuentas que no corresponden al que dice ser el propietario.

Para ayudar a mitigar posibles suplantaciones de identidad y evitar que se pueda vulnerar los medios digitales de transferencias, se implementa un segundo factor de autenticación en donde la persona es propietaria de un dispositivo físico que, entre comillas únicamente esa persona puede tener acceso.

El presente trabajo pretende aportar con ese segundo factor de autenticación, para disminuir posibles suplantaciones de identidad y ataques conocidos de a los sistemas bancarios. Mediante la generación de códigos de seguridad aleatorios que sean caducados en un tiempo predefinido y que, se permita utilizar en una sola transacción, es decir, una sola vez, disminuyendo el margen de riesgo ante posibles ataques.

En este medio la cultura de seguridad informática no es tan difundida en empresas pequeñas por lo que, con este proyecto se pretende dar a conocer la importancia de asegurar las transacciones monetarias mediante un segundo factor de autenticación, generando así la confianza y lealtad de los clientes.

3.2 Teorías y conceptos

Este apartado aborda temas relevantes para la comprensión de los procesos transaccionales a través de canales electrónicos dentro de una institución financiera. Asimismo, se examina cómo un factor de doble autenticación, como un OTP, puede reforzar significativamente la seguridad en la autenticación, facilitando la realización exitosa de flujos transaccionales, como por ejemplo una transacción monetaria.

3.2.1 OTP – contraseña de un solo uso

Uno de los métodos más utilizados como segundo factor de autenticación son las contraseñas de un solo uso (OTP), es una de las herramientas más seguras dentro de la informática, fue concebida con la finalidad de brindar una capa adicional para proteger cualquier proceso de autenticación de un usuario que utiliza un sistema en red que contiene datos sensibles como información personal, transaccional o cualquier tipo de información con carácter restringido.

Los inicios de las OTP se remontan sobre la década de 1980, debido al crecimiento y expansión de las redes de computadoras junto con los sistemas informáticos, asegurar la información comenzó a ser un tema de suma importancia, debido a que las contraseñas estáticas eran cada vez más susceptibles de ataques como phishing,

espionaje, fuerza bruta, entre otros. La idea básica detrás de este genial método es que, incluso si una contraseña estática es obtenida por un atacante, no le sirva para completar varios flujos propios de un usuario autenticado.

Una de las primeras propuestas para la generación de OTP es el método de Lamport o también conocido como Lamport´s hash, es un método basado en funciones de encriptación de una sola vía, propuesto por Leslie Lamport en 1981. Este esquema Lamport es muy seguro debido a que una vez aplicado el hash es extremadamente difícil de revertir y descubrir el valor original.

Las OTP surgen como respuesta a la necesidad de seguridad frente a los problemas que mantienen las contraseñas estáticas tradicionales, algunas de las razones se describen a continuación en la Tabla 2:

Tabla 2
Seguridad de contraseñas estáticas vs. Contraseñas de un solo uso (OTP)

Problema de seguridad	Contraseñas estáticas	Contraseñas de un solo uso (OTP)
Robo de contraseñas	Vulnerables a robo o interceptación mediante phishing, keyloggers o ataques de fuerza bruta.	Temporales y de un solo uso, disminuyen la utilidad de las contraseñas robadas.
Contraseñas débiles	Los usuarios suelen elegir contraseñas débiles o fáciles de adivinar, facilitando el acceso no autorizado.	Generadas automáticamente y, por lo general, más difíciles de adivinar.
Reutilización de contraseñas	Si una contraseña se ve comprometida, todas las cuentas asociadas podrían estar en riesgo.	Cada OTP es válida sólo para una transacción o inicio de sesión específico.

Ataques de repetición	Vulnerables a interceptación y retransmisión para obtener acceso no autorizado.	Una OTP interceptada no sería útil después de su primer uso, protegiendo contra ataques de repetición.

Con el paso del tiempo se ha mejorado y evolucionado la forma de generar OTP, se crean nuevos algoritmos más seguros, eficientes y además fáciles de utilizar por los usuarios. Métodos como Time-Based One-Time Password (TOTP) y el HMAC-based One-Time Password (HOTP), son los algoritmos más utilizados en la actualidad. TOTP es un algoritmo que se basa en el tiempo y HOTP es un algoritmo que se basa en la función criptográfica HMAC.

El uso más común y frecuente de las OTP se ve reflejado en la autenticación de usuarios dentro de un sistema informático, se utiliza como el segundo factor de autenticación para verificar su identidad antes del acceso al sistema o para realizar una transacción. Uno de los ejemplos cotidianos es el inicio de sesión a un sistema en donde el usuario ingresa su contraseña y password tradicionales para poder recibir un OTP mediante un mensaje de texto, correo o en una aplicación de autenticación.

Entre los beneficios de utilizar las OTP como segundo factor de autenticación son:

a. **Mejora de la seguridad:** Añade una capa adicional de seguridad al proceso de autenticación, dificultando el acceso no autorizado.
b. **Protección contra fuerza bruta:** Las OTP son temporales y generadas aleatoriamente, lo que dificulta adivinarlas mediante ataques de fuerza bruta.
c. **Reducción del riesgo de robo de identidad y fraude:** Las OTP incrementan significativamente la seguridad contra el uso ilícito de credenciales robadas o comprometidas, esto se debe a que pierden su validez después de su uso o un período de tiempo determinado.

d. **Prevención de ataques de repetición:** Las OTP, al ser de un solo uso, protegen contra ataques de repetición, dado que una OTP interceptada no sería útil después de su primer uso.
e. **Mayor resistencia a la suplantación de identidad (phishing):** Dado que las OTP son únicas y temporales, se reduce el riesgo de que un atacante pueda utilizar información robada en un intento de phishing.

3.2.2 Autenticación de dos factores (2FA) y multifactor (MFA)

Son técnicas utilizadas para la protección segura de la información sensible de los usuarios. Los nombres de cada técnica tanto el 2FA como el MFA sugieren que se utilicen dos y tres o más factores de autenticación respectivamente para verificar su identidad.

Para la autenticación 2FA o MFA se pueden utilizar la combinación de los siguientes factores Tabla 3:

Tabla 3
Combinación de Factores de Autenticación para un OTP Seguro

Algo que el usuario sabe	Algo que el usuario posee	Algo que el usuario es
Contraseña	Token de seguridad	Huella dactilar
PIN	Tarjeta inteligente	Reconocimiento facial
Patrón de desbloqueo	Llave de seguridad USB	Voz o iris

3.3 Propósito

Con la finalidad de dar a conocer a empresas financieras medianas y pequeñas la importancia de generar transacciones monetarias digitales mediante un proceso fiable, confiable y seguro, el propósito es aportar con herramientas tecnológicas de software que permitan autenticar y autorizar a los clientes para realizar las transacciones sin preocuparse por temas fraudulentos que puede acarrear pérdidas de dinero de cuentas bancarias.

Se plantea un proyecto que permita la generación y verificación de un Código de Seguridad OTP, mismo que será desplegado dentro de una app y será enviado para validación mediante canales seguros de internet.

3.4 Área de aplicación

Inicialmente el proyecto, está enfocado para que lo puedan utilizar instituciones financieras medianas y pequeñas integrando el segundo factor de autenticación en sus procesos transaccionales digitales, no obstante, la aplicación práctica de este trabajo puede ser implementado en todo procedimiento que sea necesario una doble autenticación para garantizar la veracidad de una transacción independientemente del tipo que sea, como por ejemplo, inicio de sesión, pago, transferencia, recuperación de claves, crédito, entre otros.

3.5 Alcance del proyecto

3.5.1 Entregables

a. Diseño y especificación de la arquitectura OTP.
b. Implementación de un POC para demostrar la funcionalidad el OTP.
c. Protocolo de pruebas y resultados obtenidos.

3.5.2 Actividades y tareas

a. Investigación sobre los requisitos específicos de las PYMEs en el ámbito financiero.
b. Diseño de la arquitectura, incluyendo algoritmos, protocolos y tecnologías involucradas.
c. Desarrollo y codificación del sistema.
d. Pruebas de seguridad y funcionalidad.
e. Elaboración de la documentación pertinente.

3.5.3 Límites del proyecto

a. El proyecto se centrará exclusivamente en la arquitectura OTP y no abordará otros métodos de autenticación.

b. El sistema se diseñará pensando en las necesidades de pequeñas y medianas instituciones financieras, y no necesariamente en grandes bancos o entidades globales.

3.5.4 Recursos

a. Herramientas y plataformas de desarrollo de software.

b. Acceso plataformas de desarrollo/pruebas.

c. Expertos en seguridad financiera y digital para consultas.

3.5.5 Criterios de aceptación

a. La arquitectura debe ser capaz de generar OTPs de forma segura y eficiente.

b. Las transacciones realizadas con el OTP deben ser seguras y verificables.

c. El OTP debe caducar de acuerdo con el tiempo establecido.

d. Debe permitir validar la funcionalidad mediante una POC.

e. Para registrar el OTP el usuario necesita iniciar sesión en sus dispositivos.

f. Las peticiones que requieran ser autenticadas lo harán mediante JWT.

3.5.6 Riesgos

a. Posibles vulnerabilidades de seguridad no detectadas.

b. Cambios en las normativas o regulaciones financieras que afecten al uso de OTPs.

c. Limitaciones tecnológicas o problemas de compatibilidad.

3.5.7 Suposiciones y restricciones

a. Se asume que las PYMEs tienen la infraestructura básica necesaria para implementar y usar la arquitectura OTP.

b. Las soluciones propuestas están basadas en tecnologías y normativas actuales y podrían necesitar actualizaciones en el futuro para adaptarse a nuevos escenarios.

3.6 Descripción general

En este apartado abordará el producto desde varias aristas, desde la importancia de la seguridad transaccional en los sistemas bancarios de usuario, la funcionalidad del producto.

3.6.1 Perspectiva del producto

El objetivo de este trabajo es la construcción de una arquitectura para la generación de OTPs que servirá como segundo factor de autenticación después del usuario y contraseña tradicionales. Cualquier usuario puede obtener la aplicación en su celular y escanear un código QR el cual deberá ser provisto por la institución financiera. A partir del registro del QR, la App generará códigos OTP para ser utilizados únicamente por su propietario. La aplicación es independiente, no está acoplada o atada necesariamente a una institución específica, por lo que su utilidad se podría extender a varios casos de uso.

La función principal es relativamente sencilla, para su uso se seguirán los siguientes pasos:

1. Obtener la aplicación en su celular
2. Escanear el código QR correspondiente
3. Un código numérico será mostrado en la pantalla del usuario

3.7 Comparación con otros métodos 2FA

La siguiente *Tabla 4* comparativa muestra las características entre tres de los métodos comúnmente utilizados como segundo factor de autenticación.

Tabla 4
Comparación de Token App, Token SMS y Token Hardware en Autenticación 2FA

Característica	Token App	Token Sms	Token Hardware
Dependencia de red	Baja	Alta	Baja
Vulnerabilidad	Baja	Media	Media
Costo	Bajo	Medio	Alta
Complejidad de uso	Baja	Baja	Alta
Tiempo de Respuesta	Bajo	Medio	Bajo

A continuación, se detalla el análisis por cada método:

3.7.1 Token app

a. **Dependencia de red:** El Token por App tiene una baja dependencia de la red, lo que significa que puede funcionar en muchos escenarios sin requerir una conexión constante.

b. **Vulnerabilidad:** Presenta una baja vulnerabilidad, lo que indica que es una opción segura frente a potenciales amenazas.

c. **Costo:** Su costo es bajo, lo que lo convierte en una opción económica para la autenticación e implementación.

d. **Complejidad de uso:** La complejidad de uso es baja, lo que sugiere que es una opción amigable y fácil de usar para los usuarios.

e. **Tiempo de Respuesta:** Ofrece un tiempo de respuesta bajo, lo que implica que es rápido.

3.7.2 Token SMS

a. **Dependencia de red:** Posee una alta dependencia de la red, lo que significa que su funcionamiento es crítico y depende del servicio de red.

b. **Vulnerabilidad:** Su nivel de vulnerabilidad es medio, lo que sugiere que, si bien es razonablemente seguro, puede ser susceptible a ciertos tipos de ataques.

c. **Costo:** El Token SMS tiene un costo medio, lo que indica que puede tener costos asociados, especialmente a gran escala.

d. **Complejidad de uso:** Tiene una baja complejidad de uso, lo que indica que es fácil de usar para la mayoría de las personas.

e. **Tiempo de Respuesta:** Su tiempo de respuesta es medio, lo que puede variar según la red y otros factores.

3.7.3 Token hardware

a. **Dependencia de red:** Al igual que el Token por App, el Token Hardware tiene una baja dependencia de la red, permitiéndole operar de manera independiente.

b. **Vulnerabilidad:** Posee una vulnerabilidad media, indicando que, si bien es más seguro que el SMS, aún podría tener ciertos puntos de exposición.

c. **Costo:** Tiene un costo alto, lo que refleja la inversión necesaria en el dispositivo físico y otros factores relacionados.

d. **Complejidad de uso:** Es el más complejo de los tres con una alta complejidad de uso, lo que podría requerir más capacitación o familiarización.
e. **Tiempo de Respuesta:** Al igual que el Token App, tiene un tiempo de respuesta bajo, lo que indica rapidez en la generación y autenticación de códigos.

3.7.4 Conclusión

El Token por App, o OTP por app, se sugiere como la opción más equilibrada al comparar estos tres métodos de autenticación. Las razones principales por las que el OTP por app se destaca son:

a. **Independencia de red:** Su baja dependencia de la red significa que puede generar códigos de autenticación incluso en ausencia de conectividad, lo que es esencial en áreas con cobertura irregular o en situaciones donde la conectividad puede ser intermitente.
b. **Seguridad:** Presenta una baja vulnerabilidad. A diferencia del Token SMS, que puede ser susceptible a ataques como la interceptación o el intercambio de SIM, las apps de token están protegidas por las medidas de seguridad inherentes del dispositivo en el que están instaladas.
c. **Costo-Efectividad:** La implementación de OTP a través de aplicaciones móviles se destaca por su economía. A diferencia del envío de mensajes de texto (SMS), este método, una vez desarrollado o implementado, no implica costos recurrentes por transacción.
d. **Facilidad de uso:** Con una complejidad baja, el OTP por app es intuitivo y amigable para el usuario, generalmente con una curva de aprendizaje corta.
e. **Rapidez:** Su tiempo de respuesta bajo asegura que los usuarios puedan obtener sus códigos de autenticación rápidamente, mejorando la experiencia del usuario y la eficiencia del proceso de autenticación.

Comparativamente, mientras que el Token Hardware puede ofrecer ventajas en términos de seguridad, su costo más alto y su mayor complejidad de uso pueden

ser barreras para su adopción. Por otro lado, el Token SMS, aunque sencillo, tiene vulnerabilidades inherentes y depende fuertemente de la red.

Por lo tanto, el OTP por App emerge como una solución de autenticación robusta, económica y amigable con el usuario.

3.8 Entorno de desarrollo

Factores esenciales a la hora de seleccionar un entorno de desarrollo que se adapte a necesidades como la facilidad de uso, la productividad e integración con varias otras herramientas necesarias para llevar a cabo este proyecto, permite visualizar IDEs y lenguajes versátiles y de prestigio en el medio.

Para el desarrollo del proyecto se utilizan herramientas Microsoft y Google, que tienen trascendencia demostrada, esto permite de cierta manera garantizar su permanencia y el respaldo del desarrollo técnico del proyecto.

Para seleccionar las herramientas de desarrollo a utilizar, básicamente se toma en cuenta los parámetros expertise y experiencia por lo que, la selección de herramientas y tecnologías se dividen en dos partes que se describen en la *Tabla 5* a continuación:

Tabla 5
Herramientas y Tecnologías Utilizadas en el Desarrollo del Proyecto

Herramienta	Versión	Frontend	Backend	Propietario	Tipo
Visual Studio Community	2022	✗	✓	Microsoft	IDE
Visual Studio Code	1.75	✓	✗	Microsoft	IDE
Flutter	3	✓	✗	Google	Framework
Dart	2.19	✓	✗	Google	Lenguaje
Javascript	ES9	✓	✗	ECMA	Lenguaje
Html	5	✓	✗	WWW	Lenguaje

3.8.1 Herramientas adicionales

a. **Postman:** Aplicación para testeo de Apis de backend.
b. **Jmeter:** Plataforma de código abierto para evaluar el rendimiento de sitios web.
c. **Smartphone:** Celular para testeo de la app.
d. **Computador:** Para ejecutar todo el ambiente de desarrollo.

3.9 Entorno productivo

Utilizando las herramientas de desarrollo actuales, la capacidad de desplegar en entornos productivos se vuelve relativamente sencilla. Aunque estas herramientas facilitan el despliegue automatizado, el compacto tamaño del proyecto permite también realizar un despliegue manual. El proceso manual es directo y no representa una carga significativa en términos de tiempo o complejidad.

El backend se puede configurar en un servidor Windows, cuyas especificaciones detalladas se tiene que evaluar de acuerdo con la transaccionalidad de la institución que implemente este proyecto. Dado que este proyecto actúa principalmente como una demostración práctica, no es esencial un alojamiento en la nube. Esta elección no solo simplifica el proceso, sino que también garantiza un control más directo sobre los datos y recursos durante la fase de demostración.

El frontend, desarrollado con Dart y Flutter, está diseñado para ser compatible con diversas tiendas de aplicaciones. Esto no solo demuestra la versatilidad de nuestra solución, sino que también subraya la facilidad con la que puede adaptarse y desplegarse en diferentes plataformas y ecosistemas.

3.10 Casos de uso

A continuación, se describen lo casos de uso para la aplicación de 2FA

Tabla 6
Registro y Generación de OTP

Título	Registro y Generación de OTP
Descripción	Es la generación de tokens en la aplicación por parte del usuario previo registro y autenticación del usuario
Actores	Usuario final
Precondiciones	✓ La aplicación debe estar instalada ✓ El usuario ha obtenido la clave secreta desde un QR generado en la web
Flujo	1. El usuario accede a la app 2. El usuario ingresa por la opción nuevo (+) 3. El usuario escanea la clave secreta desde un QR mostrado en la web 4. La app genera y muestra el OTP
Criterios de Aceptación	a. El OTP generado es válido b. El OTP generado se puede utilizar para validarlo c. El OTP cambia cada 30 segundos d. El OTP es de 6 dígitos e. El OTP tiene un límite de tiempo preestablecido

Tabla 7

Almacenamiento de claves secretas

Título	Almacenamiento de claves secretas
Descripción	Este caso indica el proceso de ingreso y almacenamiento de la clave secreta por parte del usuario.
Actores	Usuario final
Precondiciones	✓ La aplicación debe estar instalada ✓ El usuario ha obtenido la clave secreta
Flujo	1. El usuario accede a la app 2. El usuario ingresa por la opción nuevo (+) 3. El usuario ingresa la clave secreta 4. La app almacena la clave de forma segura en el store 5. La app muestra el mensaje confirmando el registro
Criterios de Aceptación	a. La clave secreta ha sido almacenada b. La cuenta ha sido creada en el dispositivo c. La app comienza a generar los otps

Tabla 8
Verificación de Otps

Título	Verificación de Otps
Descripción	Este caso describe el proceso de verificación del otp genera por el usuario en la app
Actores	Usuario final
Precondiciones	✓ La aplicación debe estar instalada ✓ El usuario a registrado su cuenta con la clave secreta ✓ El usuario a generado un otp
Flujo	1. El usuario accede a la app 2. La app muestra el otp generado 3. El usuario ingresa el otp en un sitio web independiente generado para este fin 4. El otp es validado por un servidor y el sitio web muestra el respectivo mensaje de validación correcta o incorrecta
Criterios de Aceptación	a. El otp es validado en el servidor y determina si es un código válido o no válido b. El sitio web muestra el respectivo mensaje devuelto por el servidor

3.11 Requisitos funcionales

Las funcionalidades de la app contemplan los siguientes puntos de acuerdo con el alcance del proyecto.

Tabla 9

RF-01 Almacenamiento de clave secreta de forma segura

Código	RF-01
Requisito	Almacenamiento de clave secreta de forma segura
Descripción	La clave debe ser almacenada dentro del dispositivo, únicamente la aplicación podrá tener acceso a la clave segura.

Tabla 10

RF-02 Verificación de OTP en el servidor

Código	RF-02
Requisito	Verificación de OTP en el servidor
Descripción	De manera aislada, el servidor debe validar y determinar si el código enviado es o no válido para ese usuario

Tabla 11

RF-03 Funcionamiento offline

Código	RF-03
Requisito	Funcionamiento offline
Descripción	La app debe ser capaz de funcionar sin conexión a internet, es decir, los códigos incluso el registro se podrían realizar fuera de línea

Tabla 12

RF-04 Interfaz intuitiva y de fácil uso

Código	RF-004
Requisito	Interfaz intuitiva y de fácil uso
Descripción	La aplicación de usuario final no debe ser compleja para su utilización, sencilla con textos e iconos intuitivos para que el usuario no necesite realizar algún tipo de inducción previa

3.12 Requisitos no funcionales

Tabla 13
RNF-01 Compatibilidad con Android y IOS

Código	RNF-01
Requisito	Compatibilidad con Android y IOS
Descripción	La app, debe ser compatible con Android e IOS para que a futuro pueda ser publicada en cualquiera de las tiendas.

Tabla 14
RNF-02 Facilidad de mantenimiento y escalabilidad

Código	RNF-02
Requisito	Facilidad de mantenimiento y escalabilidad
Descripción	La construcción de la arquitectura debe utilizar buenas prácticas para el desarrollo, código claro y limpio, módulos independientes para cada responsabilidad de tal forma que se pueda reutilizar de acuerdo con las necesidades

Tabla 15
RNF-03 Tiempos de respuesta menor a 2 segundos

Código	RNF-03
Requisito	Tiempos de respuesta menor a 2 segundos
Descripción	Al ser una app que permite la generación de OTP de manera offline, los tiempos de respuesta de la aplicación no deberán ser mayores a 2 segundos, de tal forma que se tenga una muy buena experiencia para el usuario

Tabla 16

RNF-04 Requisito Nivel de seguridad adecuado para proteger la clave secreta

Código	RNF-04
Requisito	Nivel de seguridad adecuado para proteger la clave secreta
Descripción	Los niveles para asegurar el almacenamiento de la clave secreta deben ser los adecuados, dentro de los cual se requiere que no se almacenen la misma clave en texto plano o con algoritmos de que sean de codificación fácilmente reversibles como base64

3.13 Modelo C4 de diagramas

A continuación, se muestran los diagramas que forman parte de la estructura del proyecto.

3.13.1 Diagrama de contexto

Este diagrama estructura de manera general los sistemas que interactúan para el proceso transaccional Figura 1.

El Cliente: Es el usuario que interactúa directamente con los sistemas, el sistema transaccional y el generador de OTP para poder autorizar una transacción.

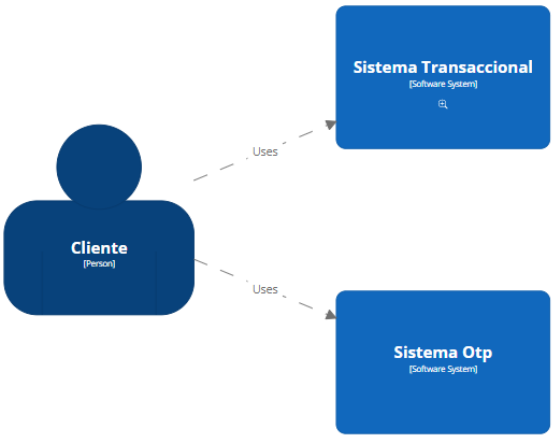

Figura 1
Diagrama de contexto

3.13.2 Diagrama de contenedor

Ya sea una aplicación web o una app móvil transaccional pueden hacer uso del generador de OTP, por lo tanto, el cliente interactúa desde cualquier plataforma.

3.13.2.1 Sistema transaccional

Un sistema transaccional de una institución financiera puede ser web o móvil, por lo que se recomendaría que las dos trabajen con el generador de OTP como segundo factor de autenticación.

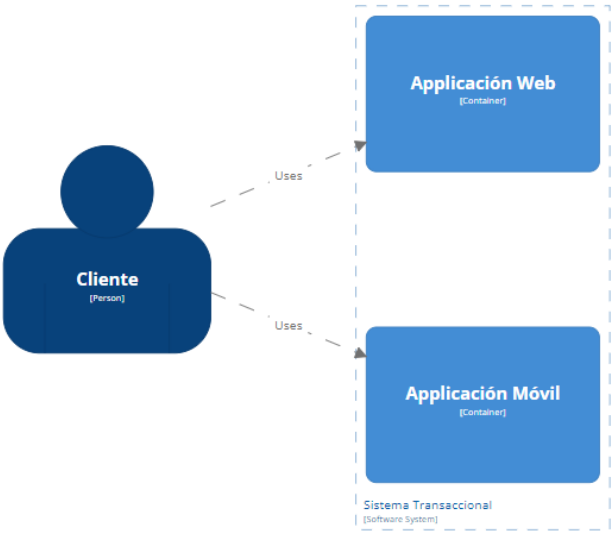

Figura 2
Sistema Transaccional (Diagrama de contenedor)

3.13.2.2 Sistema OTP

El generador de OTP, por su parte, incluye una interfaz de usuario que interactúa con otros componentes esenciales. Estos componentes incluyen servicios para generar la semilla inicial y un 'store' para el almacenamiento seguro de claves. Además, internamente, el generador trabaja con varios componentes adicionales, como clases y widgets, que son necesarios para la generación efectiva del OTP.

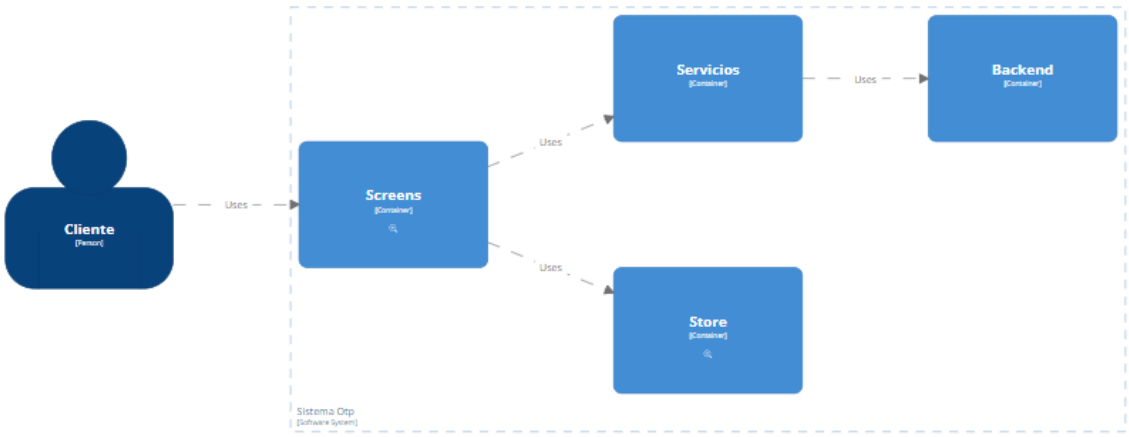

Figura 3
Sistema Otp (Diagrama de contenedor)

3.14 Prototipo

A continuación, la Figura 4 muestra el prototipo de la aplicación, un simple pero funcional diseño para mostrar la generación de OTP's.

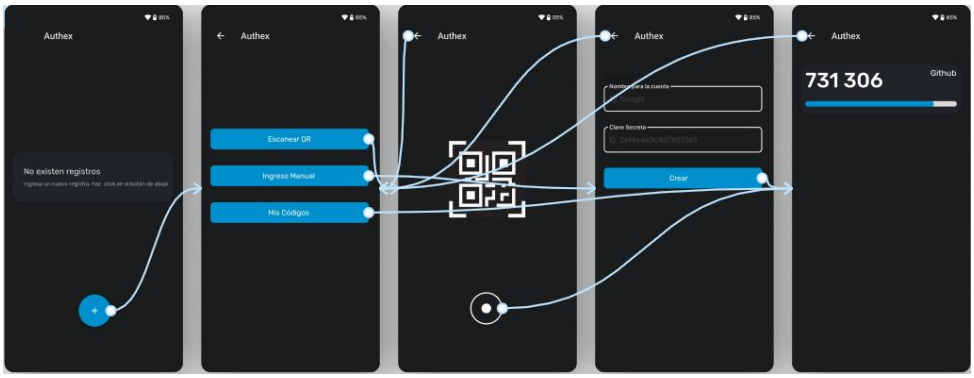

Figura 4
Prototipo

3.14.1 PANTALLA INICIAL

La pantalla de entrada Figura 5 de la aplicación muestra simplemente un mensaje para indicar al usuario que no existen registros, es decir, no existe ninguna cuenta creada para generar OTP. En la parte inferior central hay un botón que permite ingresar al menú principal de la aplicación.

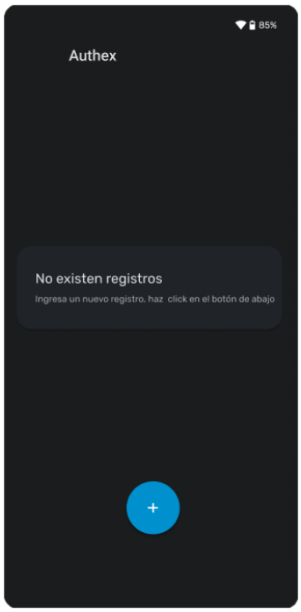

Figura 5
Pantalla Inicial

3.14.2 MENÚ PRINCIPAL

En esta pantalla Figura 6 se muestran las opciones que tiene la aplicación, en donde le permite registrar cuentas o ingresar al apartado de cuentas ya registradas. Las opciones principales se describen a continuación:

a. **Escanear QR:** Permite registrar una cuenta basada en un código QR, este código QR viene generado desde el servidor y proporciona la data para poder ejecutar el registro en el dispositivo, el QR contiene principalmente el correo asociado a la cuenta y la clave secreta a partir de la que se generará el OTP.

b. **Ingreso Manual:** Permite registrar una cuenta de manera manual, es decir el usuario tiene que ingresar el nombre de la cuenta y una clave secreta a partir de la que se generará el OTP.

c. **Mis Códigos:** Si el cliente tiene cuentas ya registradas, este botón permite dirigir a la pantalla en donde podrá ver los OTP generados para sus cuentas.

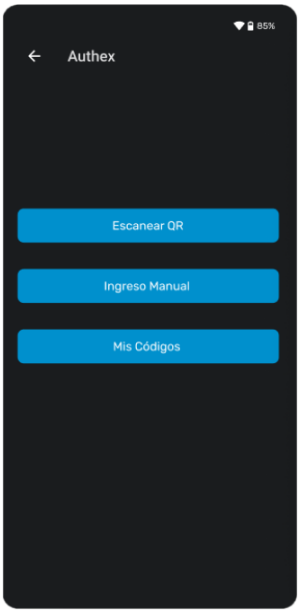

Figura 6
Menú Principal

3.14.3 Registro por código QR

Desde el menú principal Figura 6 se dirige a está pantalla Figura 7. En esta interfaz, el elemento central es la cámara, cuya función primordial consiste en leer un código QR. Este código contiene la información necesaria para el registro de la cuenta.

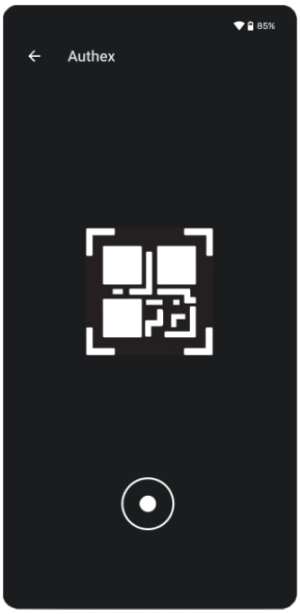

Figura 7
Registro por código QR

3.14.4 Registro manual

Esta pantalla permite Figura 8 realizar un registro manual de la cuenta, es decir, el usuario requiere digitar un nombre y una clave secreta para la cuenta.

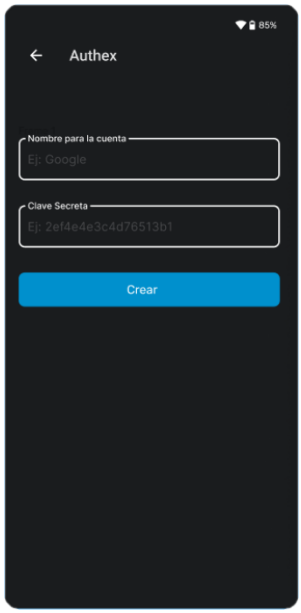

Figura 8
Registro Manual

3.14.5 Cuenta creada

Una vez que la cuenta haya sido creada, el usuario puede acceder inmediatamente para poder visualizar los códigos OTP y está listo para ser utilizado en la cuenta que esté asociado Figura 9.

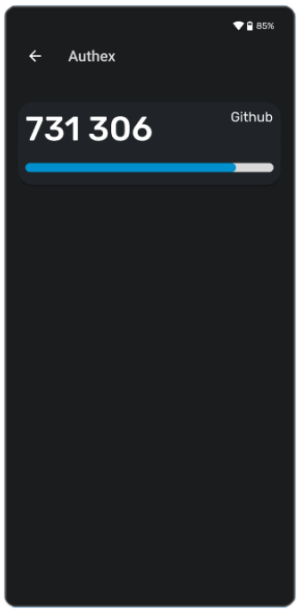

Figura 9
Cuenta Creada

3.15 Registro de decisiones de arquitectura

La arquitectura de aplicaciones de software es una parte importante que abarca la estructura y guía el desarrollo, diseño hacía la evolución del sistema. Las decisiones que se tomen en el equipo de un proyecto afectan directamente al éxito y a la mantenibilidad de un sistema en el tiempo. Dichas decisiones pueden incluir, tecnologías, estructura del proyecto, diseños, entre otros parámetros esenciales de la arquitectura del sistema.

Con la finalidad de mantener claras las decisiones del proyecto y además poder tener registrado para darle seguimiento del porqué de las decisiones, es necesario documentar estás decisiones para asegurar su compresión a medida que avanza el proyecto o para cuando otro usuario lo requiera verificarlo.

En esta sección se presenta una lista de ADR que describen las decisiones de arquitectura importantes tomadas para el proyecto. Estos ADR incluyen varios aspectos de diseño e implementación, detallando las opciones consideradas y las consecuencias de estas.

Tabla 17
ADR - Tecnología Frontend

Título	Tecnología Frontend
Fecha	2022-11-10
Versión	1
Contexto	El conocimiento adquirido en base a autoaprendizaje y luego de haber evaluado otras tecnologías, tomando en cuenta facilidad, rapidez y soporte, se decide utilizar Flutter para realizar el frontend.
Decisión	Utilizar Flutter y Dart para desarrollar el frontend
Consecuencias	Al ser un framework de Google genera principalmente confianza para el desarrollo. Además de su rendimiento, permite crear apps atractivas hacia el cliente.
Status	Aprobado

Tabla 18
ADR - Algoritmo de hash

Título	Algoritmo de hash
Fecha	2022-11-10
Versión	1
Contexto	Los algoritmos de hash son sumamente importantes para la generación de información segura, y utilizar un hash seguro en este tipo de aplicaciones es determinante para garantizar la seguridad.
Decisión	Utilizar hmacsha256 para prevenir ataques de colisión asegurando nuestra clave secreta.

Consecuencias	Generación de OTP único para cada cliente, esto garantiza un factor de doble autenticación robusto.
Status	Aprobado

Tabla 19

ADR - OTP Basado en el tiempo

Título	OTP Basado en el tiempo
Fecha	2022-11-10
Versión	1
Contexto	Debido a que el factor tiempo es una variable que obviamente no se repite, se realizará la generación del OTP basado en el tiempo como una de las variables.
Decisión	Generar OTP basado en el tiempo.
Consecuencias	Obtener OTP diferentes para cada transacción que el usuario realice el usuario, garantizando la identidad del mismo usuario.
Status	Aprobado

Tabla 20

ADR - Almacenamiento de claves en el celular

Título	Almacenamiento de claves en el celular
Fecha	2022-11-10
Versión	1
Contexto	Para evitar que la clave secreta sea vulnerada, se debe almacenar la clave en sistemas de almacenamiento seguros y certificados dentro del dispositivo.
Decisión	Utilizar el sistema de almacenamiento Keystore del dispositivo
Consecuencias	Mantener la clave protegida encriptada. Accesos restringidos a la clave solo por la aplicación autorizada.
Status	Aprobado

Tabla 21

ADR - Comunicación segura entre el cliente y el servidor

Título	Comunicación segura entre el cliente y el servidor
Fecha	2022-11-10
Versión	1
Contexto	El transporte de información entre el cliente y servidor debe hacerse mediante https, debe protegerse los datos de manera que no viajen datos en texto claro.
Decisión	Utilizar el protocolo HTTPS y token JWT para garantizar el transporte seguro de los datos
Consecuencias	Brinda una capa más de seguridad al esquema planteado para la generación de OTP.
Status	Aprobado

Tabla 22

ADR - Generación Offline de OTP

Título	Generación Offline de OTP
Fecha	2022-11-10
Versión	1
Contexto	La información que viaja mediante la red de internet es susceptible de ataques informáticos, mucho más aún si no viaja con las medidas de seguridad necesarias. Al plantear el uso del OTP como un segundo factor de autenticación es mejor generarlo de manera desconectada.
Decisión	Generar los OTP de modo desconectado de la red de internet, para mitigar el riesgo de captura en el medio del OTP.
Consecuencias	El OTP generado no tiene que ser transportado desde un servidor hacía un cliente. El OTP no se almacena en ningún lugar, por lo que mejora la seguridad.
Status	Aprobado

Tabla 23

ADR - Clave Secreta Generada Automáticamente

Título	Clave Secreta Generada Automáticamente
Fecha	2022-11-10
Versión	1
Contexto	La clave secreta debe ser robusta y aleatoria, por lo que no se recomienda realizar mediante algoritmos o frameworks validados, para que el cliente no inserte una clave débil.
Decisión	Para este proyecto, se integra Identity framework core de .NET de tal manera que el servidor me entregue una clave secreta de acuerdo con el usuario.
Consecuencias	Mantiene una clave aleatoria y robusta. El usuario no tiene que almacenar la clave ni recordarla.
Status	Aprobado

Tabla 24

ADR - Caducidad de OTP

Título	Caducidad de OTP
Fecha	2022-11-10
Versión	1
Contexto	Para evitar que un OTP pueda ser utilizado de manera indefinida, se debe de dar un tiempo de vida, por ello es importante basarnos en el factor temporal.
Decisión	Generar los OTP cada 30 segundos
Consecuencias	Los OTP caducan cada 30 segundos. Los OTP no pueden ser reutilizados por otras personas.
Status	Aprobado

3.16 Diagrama de arquitectura

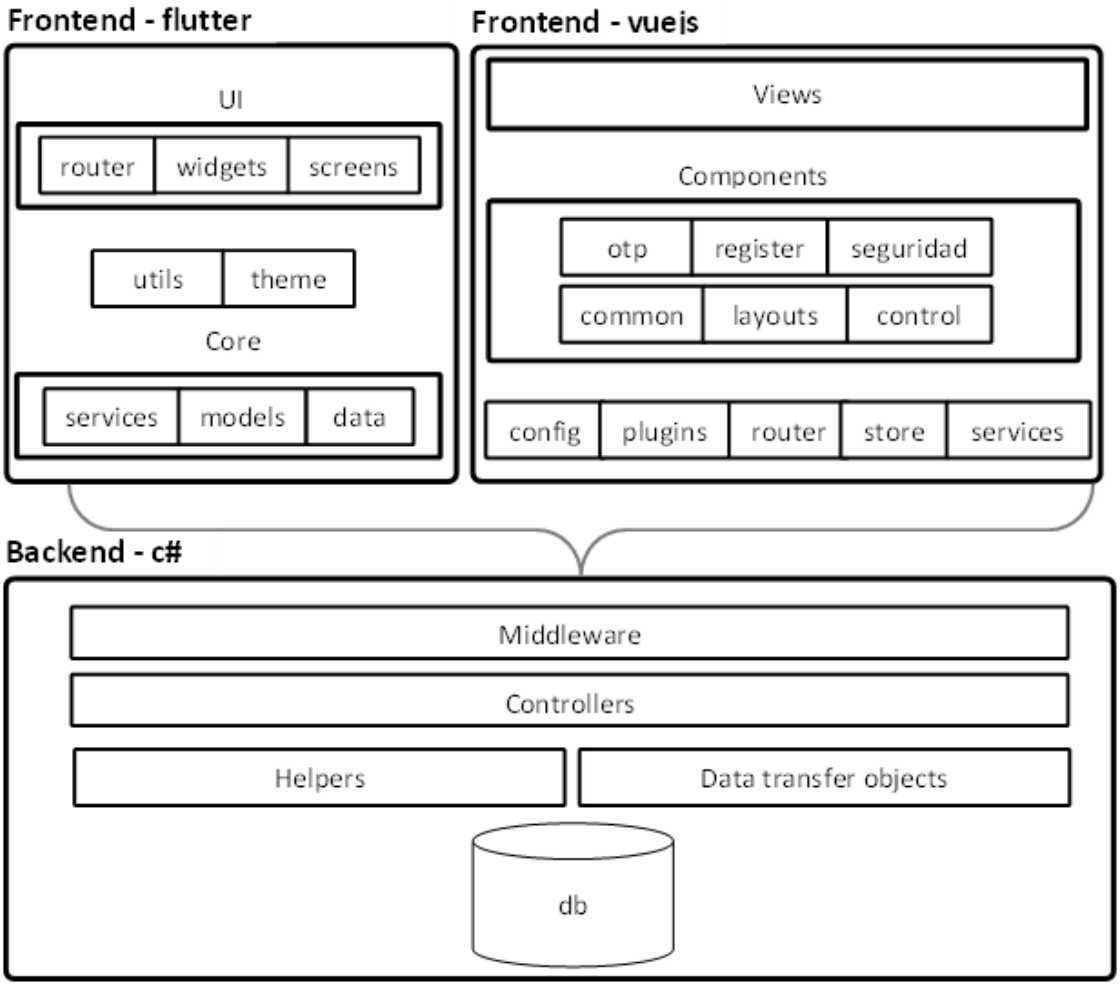

Figura 10

Diagrama de arquitectura

3.16.1 Descripción de la arquitectura

La arquitectura del sistema Figura 10 se ha diseñado como un paradigma multicapa que integra principios de microservicios y una estructura en capas que promueve la separación de responsabilidades, la flexibilidad y el mantenimiento eficiente del código.

La arquitectura promueve el uso de código limpio, mejorando la legibilidad y la mantenibilidad del sistema. El patrón repositorio encapsula la lógica de acceso a datos, facilitando su posible sustitución y la realización de pruebas unitarias. La inyección de dependencias se implementa para minimizar el acoplamiento entre

clases, aumentando la flexibilidad y simplificando la gestión de dependencias. La orientación a objetos se aplica para encapsular el comportamiento dentro de clases bien definidas, aprovechando conceptos como la herencia y el polimorfismo para mejorar la reutilización del código.

En conjunto, esta arquitectura multifacética permite la escalabilidad y evolución del sistema, manteniendo un alto nivel de cohesión y una baja dependencia entre sus componentes, lo que resulta en un sistema robusto y adaptable a cambios futuros y a la integración con otras tecnologías o frameworks.

3.16.2 Backend

La arquitectura del proyecto no obedece estrictamente a una arquitectura específica, no obstante, está basada en varios enfoques y patrones arquitectónicos, que combinados con buenas prácticas y principios de desarrollo, permiten tener una estructura en capas independientes la una de la otra.

En el lado del backend se mantiene una arquitectura enfocada en servicios y separando las responsabilidades en cada uno de los métodos para atender peticiones de clientes. Las Apis se encargan de responder las peticiones y así mismo persistir la información en la base de datos mediante el patrón repositorio embebido en el ORM.

A continuación, se detalla la función de cada componente de backend dentro del esquema arquitectónico:

Middleware: Capa intermedia que maneja las solicitudes HTTP, aplicando lógica adicional como autenticación y autorización.

Controllers: Encargados de recibir las entradas de las solicitudes, interactuar con los modelos o servicios necesarios y devolver las respuestas adecuadas.

Helpers: Colección de clases y métodos auxiliares que proporcionan funcionalidades transversales como formateo de datos, validación y operaciones comunes.

Data Transfer Objects: Objetos que definen la estructura de los datos que se transfieren entre el cliente y el servidor, promoviendo la independencia entre la capa de datos interna y la que se expone a los clientes.

Database (db): Representa la base de datos utilizada para la persistencia de datos, que interactúa con el backend a través de la capa de acceso a datos.

3.16.3 Frontend

Para la funcionalidad y demostración, el proyecto cuenta con dos frontend independientes, el uno es la aplicación móvil generadora de OTP y el otro una aplicación web que permite mostrar un código QR para que sea leído y registrado por la aplicación móvil.

El frontend web permite la autenticación con un usuario específico, de tal forma que, al solicitar el código QR se genere mediante un identificador único para cada usuario.

El frontend móvil o app, permite la captura y almacenamiento de la información generada en el código QR, mediante la cámara del dispositivo se realiza la lectura del QR y se almacena de manera segura en el keystore del dispositivo para posteriormente recuperar y proceder a generar los códigos OTP.

A continuación, se detalla la función de cada componente de frontend dentro del esquema arquitectónico:

Frontend - Flutter:

UI: Implementa la interfaz de usuario, compuesta por router, widgets y screens, que definen la navegación, los elementos interactivos y las pantallas respectivamente.

Utils y Theme: Proporcionan utilidades generales y definiciones de temas para garantizar la consistencia y reusabilidad en el diseño de la interfaz de usuario.

Core: Contiene los servicios esenciales (services), modelos de datos (models) y capa de persistencia (data), asegurando una separación clara entre la lógica de negocio y la interfaz de usuario.

Frontend - Vue.js:

Views: Contiene los componentes visuales y las views que presentan los datos y capturan la interacción del usuario.

Components: Incluye elementos reutilizables como OTP, register, y componentes relacionados con la seguridad (seguridad), así como common, layouts, y control.

Config, Plugins, Router, Store, Services: Estos directorios encapsulan la configuración global, plugins, gestión de rutas, estado global de la aplicación (Vuex store) y servicios de comunicación con el backend.

3.17 Estructura del proyecto

Para la implementación de este proyecto y poder demostrar la funcionalidad de este, de manera general se divide en dos partes esenciales, la primera corresponde a la aplicación móvil, la cual se encarga de la generación de códigos OTP y la segunda parte corresponde a la aplicación de servidor que se encargará de la validación del código OTP que se generó en el móvil.

La funcionalidad para el proceso completo de este proyecto, contempla varios pasos en el que se simula la autorización de transacciones mediante la verificación del OTP de lado del servidor, siendo aislado del generador de OTP que se encuentra de lado del cliente. Para esto tanto el front como el back se subdividen también en dos partes, el backend trabaja un proyecto de código en conjunto con una base de datos para el almacenamiento de tokens de usuarios. El frontend por su lado ha sido diseñada una aplicación móvil que se encarga de administrar los OTP del usuario/cliente y, una aplicación web que utilizada para registrar y validar los OTP de un cliente predeterminado.

3.17.1 Estructura del backened

1. **DataTransferObjects** (Capa de clases de propiedades get y set)
   1.1 GeneralDTO (Clases de propósito y uso general dentro del backend)
   1.2 IdentityDTO (Clases que se usan para la administración de usuarios)
2. **TransactionalApi** (Capa de Apis)
   2.1 ConfigSections (Clases de propiedades que leen desde settings json)
   2.2 Controllers (Clases de controladores que interactúa con el frontend)
   2.3 Datos (Extensión de propiedades de IdentityDbContext)
   2.4 DependencyInjection (Configuración para el contenedor de inyección de dependencias)
   2.5 Helpers (Métodos de ayuda)
   2.6 Middleware (Componente para validar el request antes de permitir el acceso a los servicios)
   2.7 Migrations (Migraciones para la base de datos)
   2.8 Models (Extensiones personalizadas para la migración)
   2.9 appsettings.json (Configuración de variables globales)

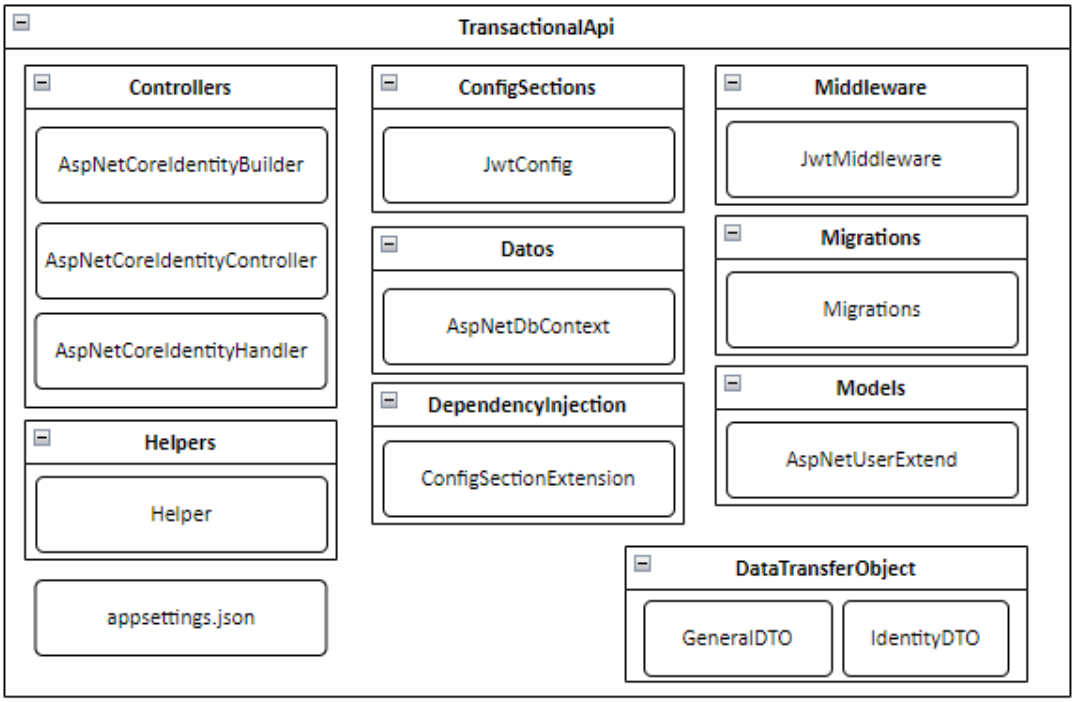

Figura 11
Estructura de backend

3.17.2 Estructura del frontend – app móvil

1. **lib** (Directorio principal de todo el desarrollo personalizado)
   - **1.1 core** (Directorio para comunicación con el backend)
     - **1.1.1 models** (Contiene clases de estructuras de datos)
     - **1.1.2 services** (Servicios de comunicación con el backend)
       - **1.1.2.1 api** (Configuración central para los request)
       - **1.1.2.2 authentication** (Servicios para autenticación de usuario)
   - **1.2 theme** (Configuración del tema de la aplicación)
   - **1.3 ui** (Capa para la administración de interfaces de usuario)
     - **1.3.1 router** (Administración de rutas de la aplicación)
     - **1.3.2 screens** (Pantallas de la aplicación)
       - **1.3.2.1 authentication** (Pantallas de autenticación de usuario)
       - **1.3.2.2 home** (Pantallas del menú principal)
       - **1.3.2.3 manualRegister** (Pantalla para registro por código)
       - **1.3.2.4 otpGenerator** (Pantalla de generación de OTP)
       - **1.3.2.5 qrRegister** (Pantalla para registro por QR)
     - **1.3.3 widgets** (Widgets personalizados de uso común)
   - **1.4 utils** (Utilidades generales para la aplicación)
   - **1.5 pubspec.yaml** (Definiciones de configuración y estructura del proyecto)

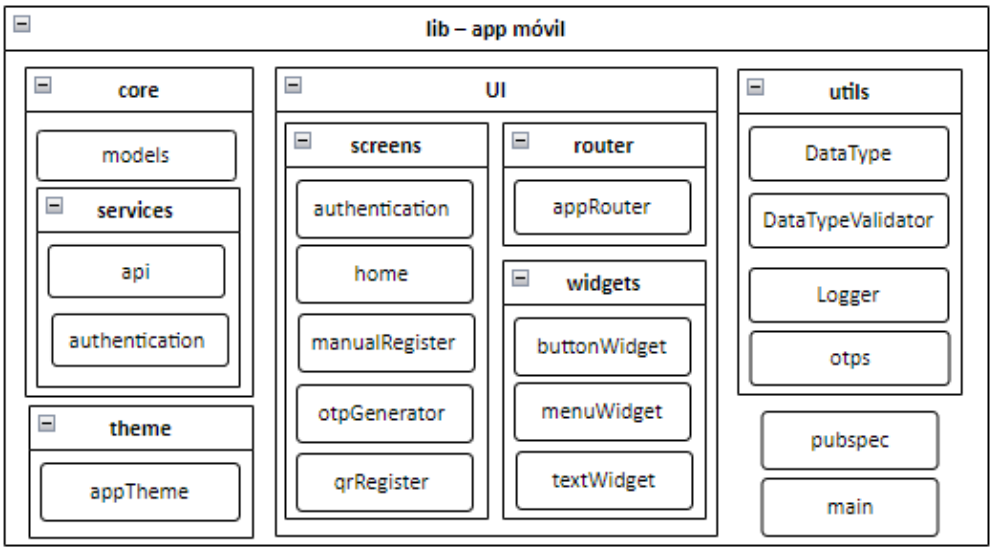

Figura 12
Estructura frontend móvil

3.17.3 Estructura del front – app web

1. **src** (Directorio principal de todo el desarrollo personalizado)
   - **1.1 assets** (Directorio de recursos estáticos)
   - **1.2 components** (Todos los componentes personalizados)
     - **1.2.1 common** (Componentes comunes)
     - **1.2.2 control** (Controles/Widgets/Componentes genéricos)
     - **1.2.3 layouts** (Plantillas de la aplicación)
     - **1.2.4 otp** (Componente del OTP)
     - **1.2.5 register** (Componente de registro)
     - **1.2.6 securities** (Componentes de autenticación)
   - **1.3 config** (Configuración de endpoints centrales de la aplicación)
   - **1.4 css** (Hojas de estilo)
   - **1.5 mixins** (Métodos reutilizables)
   - **1.6 plugins** (Características adicionales para los componentes)
   - **1.7 qrcode** (Archivos js para la generación del qr)
   - **1.8 router** (Configuración y definición de rutas)
   - **1.9 services** (Archivos js que centralizan las peticiones para el servidor)
   - **1.10 store** (Almacén centralizado para gestionar el estado)
   - **1.11 views** (Componentes de pantallas del cliente)
   - **1.12 App** (Componente raíz de la aplicación)
   - **1.13 main** (Configuraciones para inicializar la aplicación)

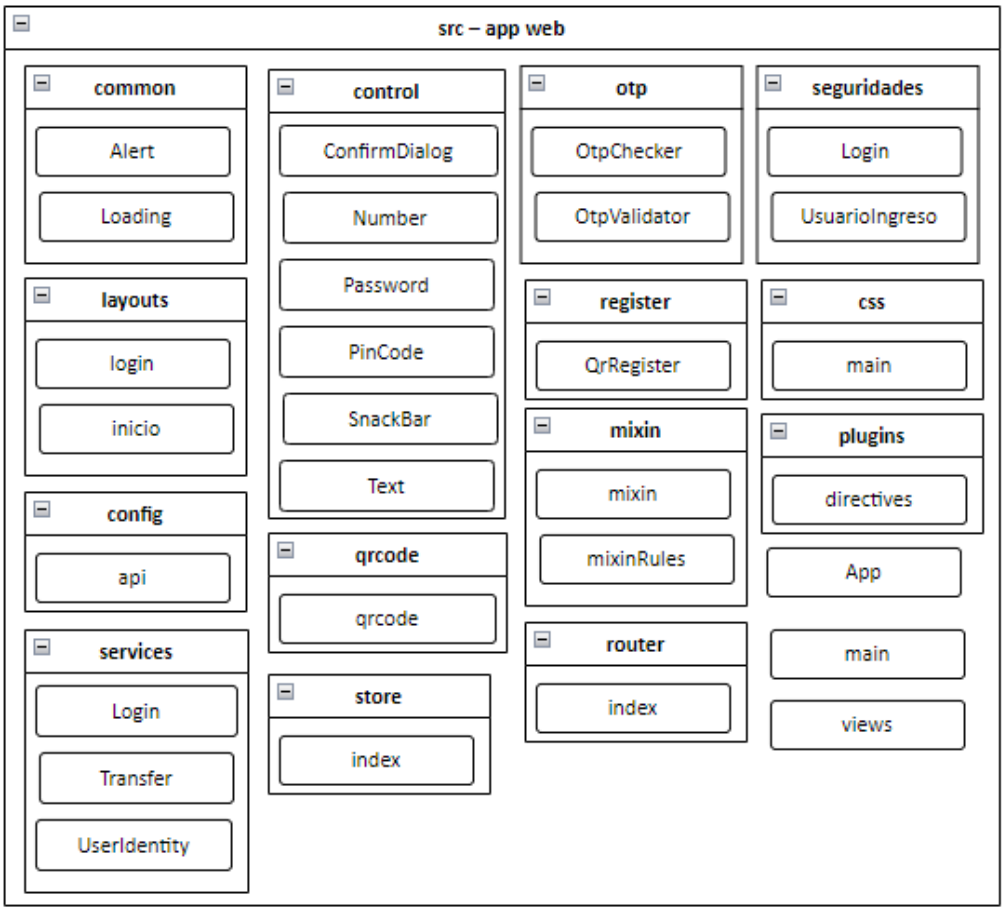

Figura 13
Estructura frontend web

3.18 Tecnologías utilizadas

La elección de tecnologías apropiadas es uno de los factores fundamentales a tomar en cuenta para garantizar la eficiencia, escalabilidad y mantenibilidad de las aplicaciones. No solo se trata de seleccionar herramientas que sean populares o estén en tendencia, sino de elegir aquellas que mejor se alineen con los requisitos del proyecto y las habilidades del equipo de desarrollo.

En el contexto de este proyecto, se buscó combinar tecnologías robustas y versátiles que se adapten tanto al desarrollo web como al móvil, garantizando así una experiencia homogénea y eficiente para los usuarios finales y una base de código mantenible para los desarrolladores.

A continuación, se describen las principales tecnologías seleccionadas y empleadas en la realización de este proyecto.

3.18.1 Backend

a. **Csharp (c#):** Un lenguaje de programación comúnmente conocido como “si sharp”, es uno de los lenguajes más populares para desarrollo de Apis, por su utilidad y facilidad de uso en conjunto con sus frameworks ha sido incluido en este proyecto para la creación de servicios.
b. **Net Core (6):** Es un framework de desarrollo multiplataforma con capacidades y rendimiento mejorado con respecto a versiones anteriores, permite trabajar en Windows, macOS y Linux, muy versátil tanto para plataformas On Premise o nube.
c. **AspnetCore Identity:** Es un framework de autorización y autenticación basado en el patrón de diseño de Identidad y Acceso Controlado (IAM), facilita la autenticación y administración de usuarios con roles, tokens, notificaciones entre otros además de su versatilidad para acoplarse con diferentes bases de datos para el almacenamiento de información.

3.18.2 Frontend

d. **Dart:** Es un lenguaje de programación utilizado principalmente para trabajar con el framework Flutter, posee todas las características de los lenguajes de programación orientados a objetos y permite desarrollar aplicaciones móviles nativas para Android e IOS.
e. **Flutter:** Es un framework relativamente nuevo pero popularizado gracias a su enfoque en la creación de Apps atractivas y fluidas tanto en Android como en IOS. Como características principales se destaca por su alto rendimiento, desarrollo de aplicaciones nativas, hot reload y al ser soportado por google posee una amplia comunidad y muy buen soporte.
f. **Vuejs:** Es un framework moderno de javascript que permite la creación de aplicaciones web, basado en un sistema de componentes con características de fácil utilización como su reactividad, directivas, store, entre otros. Cuenta además con una amplia comunidad activa y también con un excelente

ecosistema de complementos y bibliotecas que facilitan la integración con muchos frameworks.

g. **Vuetify:** Es una biblioteca de componentes de interfaces de usuario específicamente para vuejs. Permite la rápida creación de pantallas atractivas y fluidas, sus principales características son: un conjunto de componentes predefinidos, diseño responsivo, temas predefinidos y una muy buena documentación de fácil entendimiento.
h. **Crypto:** Es una biblioteca de javascript enfocada en la protección de datos en aplicaciones web. Su función principal es permitir el uso de algoritmos criptográficos como AES, SHA y HMAC para cifrar y descifrar información.
i. **Flutter secure storage:** Es un paquete de flutter que permite el almacenamiento y recuperación segura de datos sensibles dentro de una aplicación de flutter.

3.19 Flujos de datos y API

Brindar una experiencia fluida y dinámica al usuario final es fundamental dentro de las aplicaciones que brindan servicios de transferencias financieras. En este sentido la interacción entre las aplicaciones frontend con diferentes servicios y fuentes de datos es esencial, esta interacción, en gran medida, se lleva a cabo mediante el uso de APIs y flujo de datos.

La importancia de las APIs (Interfaces de Programación de Aplicaciones) en el ámbito financiero se conjuga con la seguridad en el transporte de información del cliente, ofreciendo un “puente” de comunicación entre las capas de frontend y los servidores o servicios backend. Mediante una API se permite el flujo de solicitud, recepción y envío de datos de manera eficiente y segura. En el contexto del proyecto, gracias a las APIs, por ejemplo, un usuario puede realizar una transferencia en su cuenta en línea de manera segura, obtener y enviar dinero tiempo real, con la tranquilidad que siempre se tendrá una verificación extra de su identidad.

Por otro lado, el flujo hace referencia a cómo la información circula entre las diferentes capas de la aplicación. Esto incluye cómo se realizan las peticiones, como se almacena, como viaja y cómo se administra el adecuado almacenamiento de los datos. En el ámbito financiero una adecuada gestión de datos es esencial para asegurar la confidencialidad de estos.

En conclusión, tanto las API como la gestión de flujos de datos son componentes cruciales en el diseño y funcionamiento de aplicaciones frontend modernas. Estos elementos aseguran que las aplicaciones no solo sean estéticamente agradables, sino también funcionales, eficientes y centradas en el usuario.

3.20 Consideraciones de seguridad

La seguridad es uno de los pilares esenciales en el desarrollo de aplicaciones web y móviles, especialmente cuando se trata de interactuar con datos y comunicarse con servicios externos a través de API. Las siguientes son algunas de las consideraciones clave que todo desarrollador frontend debe tener en cuenta:

a. **Comunicación Segura:** Siempre se debe utilizar protocolos seguros como HTTPS en lugar de HTTP. HTTPS cifra la comunicación entre el cliente y el servidor, lo que ayuda a proteger los datos del usuario de posibles interceptaciones.
b. **Validación de Entradas:** Las entradas del usuario (como formularios) siempre deben ser validadas y sanitizadas antes de ser procesadas. Esto ayuda a prevenir ataques como la inyección de código.
c. **Tokens y Autenticación:** Al interactuar con API, es esencial asegurarse de que existan mecanismos de autenticación robustos. Los tokens, como JWT (JSON Web Tokens), son ampliamente utilizados para validar la identidad y garantizar que solo los usuarios autorizados puedan acceder a ciertos recursos.
d. **Manejo de Errores:** En lugar de mostrar mensajes de error detallados al usuario, es recomendable ofrecer respuestas genéricas para no revelar información potencialmente sensible sobre la estructura o funcionamiento interno de la aplicación.

e. **CORS (Cross-Origin Resource Sharing):** Es una medida de seguridad implementada en navegadores web que restringe las solicitudes de recursos a dominios desconocidos. Al trabajar con APIs, es fundamental configurar adecuadamente las políticas de CORS para permitir solo las conexiones necesarias y seguras.

f. **Actualizaciones y Mantenimiento:** Las amenazas de seguridad evolucionan constantemente. Es crucial mantener la aplicación y todas sus dependencias actualizadas para protegerse contra vulnerabilidades conocidas.

g. **Limitación de Peticiones:** Implementar un límite en la cantidad de peticiones que un usuario puede hacer en un período determinado (rate limiting) puede ayudar a prevenir ataques de fuerza bruta o DDoS.

h. **Almacenamiento Seguro:** Si se almacena información en el lado del cliente (por ejemplo, en local storage), es vital garantizar que esta información no sea sensible o, si lo es, que esté adecuadamente cifrada.

4. Resultados y discusión

En el entorno financiero actual, la seguridad de las transacciones en línea es una preocupación primordial. A medida que los métodos tradicionales de autenticación y verificación se enfrentan a crecientes amenazas cibernéticas, surge la necesidad de implementar soluciones más robustas y seguras. En este contexto, el uso de códigos OTP (One-Time Password) se ha consolidado como una herramienta esencial para garantizar la autenticidad de las operaciones y proteger los activos de los clientes.

El objetivo principal de este proyecto es desarrollar una arquitectura para la generación de códigos OTP, con el fin de asegurar las transacciones monetarias de un cliente. Para lograrlo, se llevó a cabo un análisis del estado del arte de los sistemas OTP. Se busca diseñar una arquitectura que garantice la unicidad, confidencialidad e integridad de estos códigos. Además, se abordará el desarrollo de una aplicación móvil que integre esta arquitectura, así como la creación de una API que facilite su implementación en diversos proyectos.

En cuanto a las pruebas tecnológicas, se realizan evaluaciones específicas para asegurar que la arquitectura propuesta cumpla con los estándares de seguridad y desempeño esperados. A través de estas pruebas, se busca no solo validar la efectividad del sistema, sino también identificar áreas de mejora y optimización.

4.1 Checklist de pruebas funcionales

Tabla 25
Checklist de Pruebas funcionales

Prueba	Resultado esperado	Estado
Generación de código Qr	Mostrar código QR en pantalla para ser escaneado desde la aplicación	Validado
Escaneo de QR desde la aplicación móvil	La aplicación registra los datos del QR en el equipo móvil	Validado
Generación de OTP en la aplicación móvil	La aplicación genera OTP automáticamente	Validado
Ingresar en la Web un código cualquiera, que no sea el generado en la App	Al enviar a validar el OTP ingresado, el resultado es un mensaje de “No se pudo configurar el segundo factor de autenticación..!!“	Validado
Ingresar en la Web un código OTP generado en la APP.	Al enviar a validar el OTP ingresado, el resultado es un mensaje de “Autenticador configurado correctamente..!!“	Validado
Verificar que el OTP se genere cada 30 segundos	Un nuevo OTP se genera cada 30 segundos en la app móvil	Validado
Eliminar el registro de generación de OTP de la app	Se elimina el registro de la App, los OTP ya no aparecen en la App	Validado

4.2 Checklist prueba de concepto – simular transferencia

Tabla 26
Checklist prueba de concepto – simular transferencia

Prueba	Resultado esperado	Estado
Iniciar sesión en la Web.	El sistema le permite ingresar a la pantalla principal.	Validado
Ingresar a la pantalla de transferencias y llenar los datos.	Se carga la pantalla y solicita la información requerida para simular la transferencia.	Validado
Ingresar en la Web un código cualquiera, que no sea el generado en la App.	Al enviar a validar el OTP ingresado, el resultado es un mensaje de “Otp incorrecto..!!“.	Validado
Ingresar en la Web un código OTP generado en la APP.	Al enviar a validar el OTP ingresado, el resultado es un mensaje de “Transferencia exitosa..!!“.	Validado

4.3 Pruebas técnicas

Con la finalidad de obtener una evaluación detallada del rendimiento y la resistencia del sistema bajo diferentes condiciones de carga, se emplearon dos herramientas especializadas: Postman y JMeter. Estas pruebas permiten anticipar problemas potenciales en situaciones de producción real y garantizar la satisfacción del usuario.

4.3.1 Carga secuencial

- **Herramienta:** Postman
- **EndPoint**: “/api/AspNetCoreIdentity/TestGenerarQrPorUsuario”

Las pruebas secuenciales consisten en enviar una serie determinada de peticiones al sistema de manera consecutiva, incrementando gradualmente la cantidad en cada serie.

Los resultados son los siguientes:

Tabla 27
Pruebas de carga con peticiones secuencial

Peticiones	Duración	Tiempo Promedio	Tiempo Máximo	Tiempo Mínimo	Figura
10	2s 247ms	78ms	184ms	35ms	Figura 14
50	6s 698ms	6ms	583ms	22ms	Figura 15
100	13s 447ms	61ms	342ms	24ms	Figura 16
250	37s 605ms	81ms	486ms	23ms	Figura 17
500	1m 6s	56ms	446ms	21ms	Figura 18
1000	2m 24s	69ms	638ms	21ms	Figura 19

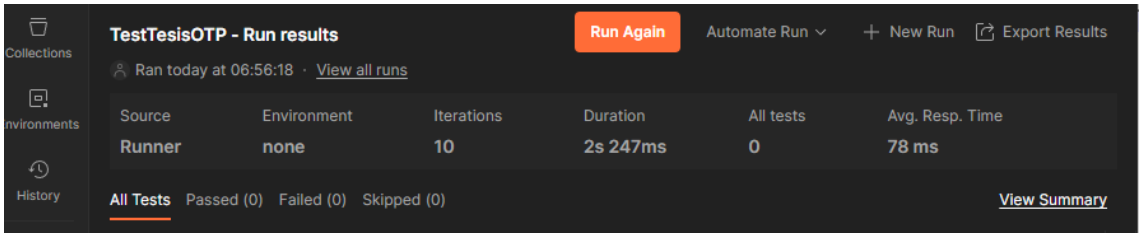

Figura 14
Prueba con 10 peticiones

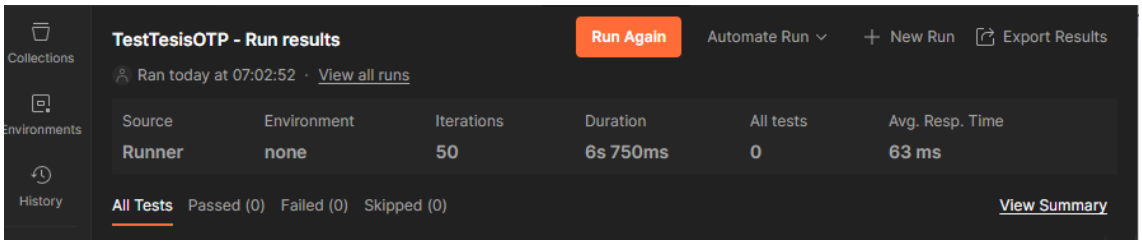

Figura 15
Prueba con 50 peticiones

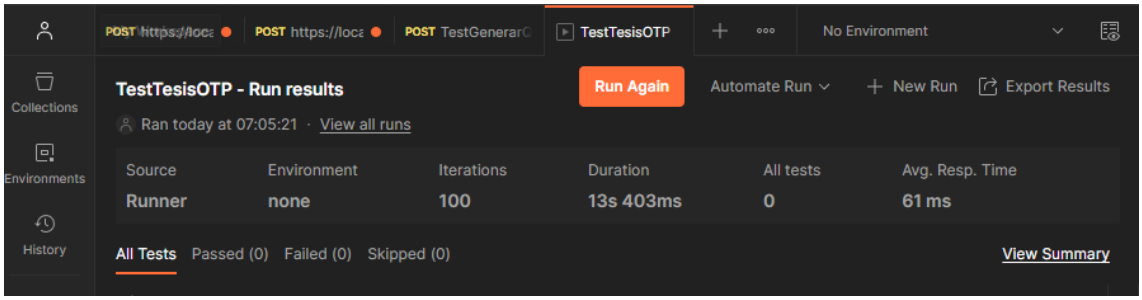

Figura 16

Prueba con 100 peticiones

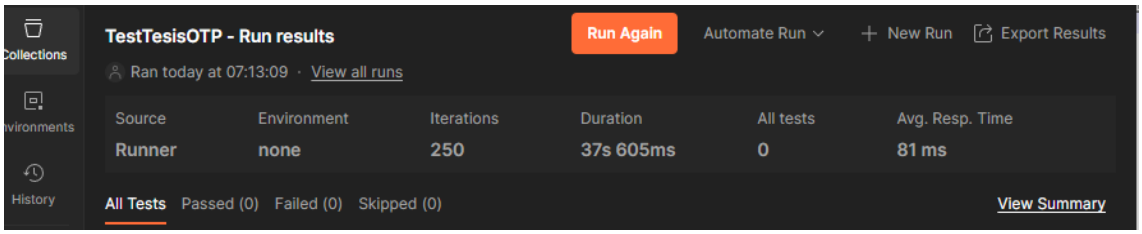

Figura 17

Prueba con 250 peticiones

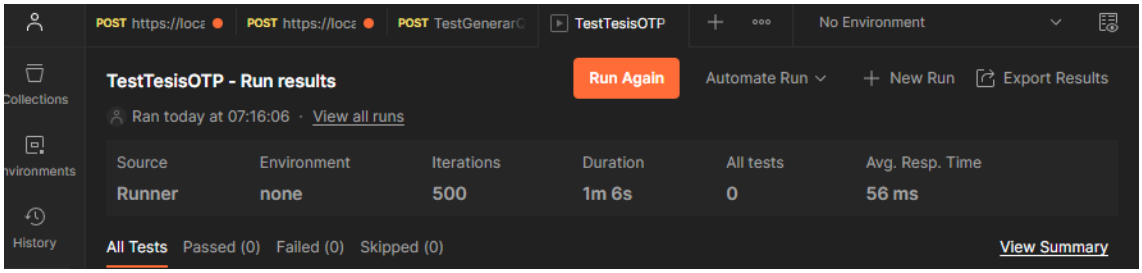

Figura 18

Prueba con 500 peticiones

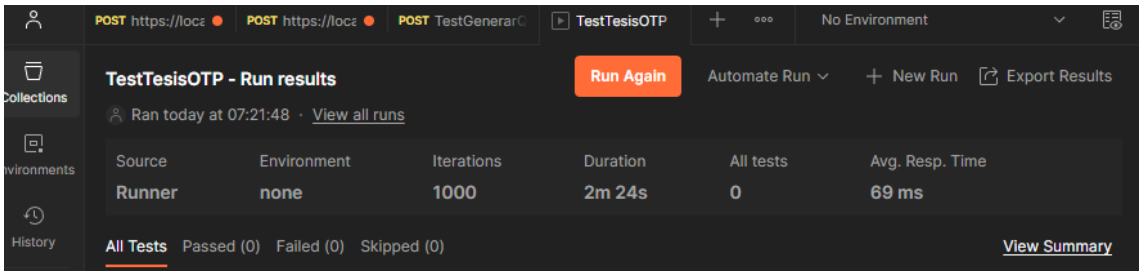

Figura 19

Prueba con 1000 peticiones

Como se puede notar, a medida que el número de peticiones aumenta, se puede observar cómo afecta al tiempo promedio de respuesta, así como identificar los tiempos máximos y mínimos de respuesta. Por ejemplo, con 10 peticiones, el

sistema tomó en promedio 78ms para responder; sin embargo, al aumentar las peticiones a 5000, el tiempo promedio disminuyó a 28ms. Esto puede deberse a diversos factores como el caching, la optimización del sistema ante cargas mayores, o particularidades en el manejo de conexiones. Es fundamental estudiar estos resultados en detalle para entender completamente el comportamiento del sistema y realizar las optimizaciones adecuadas.

4.3.2 Carga concurrente

- **Herramienta:** Jmeter
- **EndPoint:** "/api/AspNetCoreIdentity/TestGenerarQrPorUsuario"

Para evaluar la respuesta del sistema frente a una carga recurrente, se empleó la herramienta Jmeter. Este tipo de pruebas se enfoca en el envío constante y repetido de peticiones durante un periodo de tiempo, en lugar de incrementar el número de peticiones de forma secuencial. El objetivo es observar cómo el sistema maneja múltiples peticiones que se reciben en un patrón recurrente, simulando un escenario donde múltiples usuarios hacen solicitudes constantes al endpoint.

Los resultados obtenidos son:

Tabla 28
Pruebas de carga QR con peticiones concurrentes

Usuarios	Duración	Tiempo Promedio	Tiempo Máximo	Tiempo Mínimo	Figura
100	3s	2286ms	2539ms	486ms	Figura 20
250	8s	6626ms	7251ms	418ms	Figura 21
400	9s	7810ms	8685ms	335ms	Figura 22

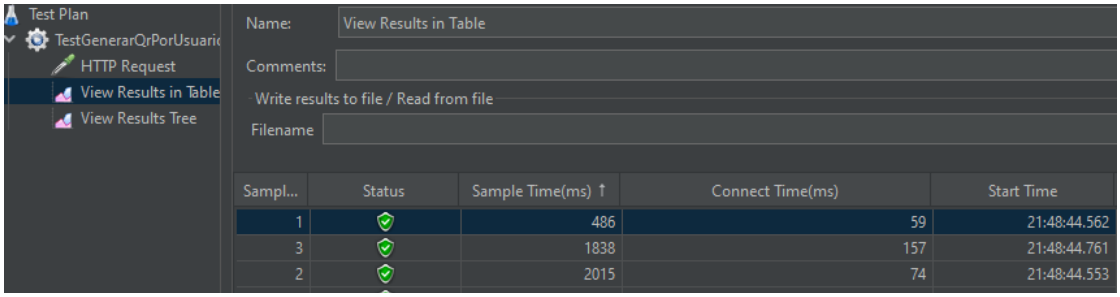

Figura 20

Carga concurrente QR 100 usuarios

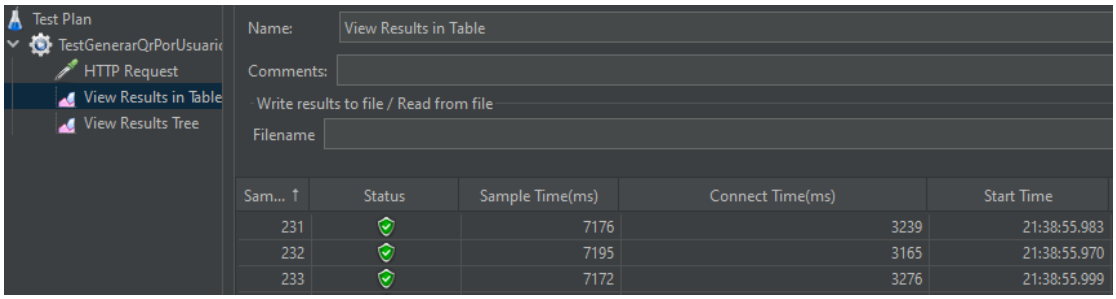

Figura 21

Carga concurrente QR 250 usuarios

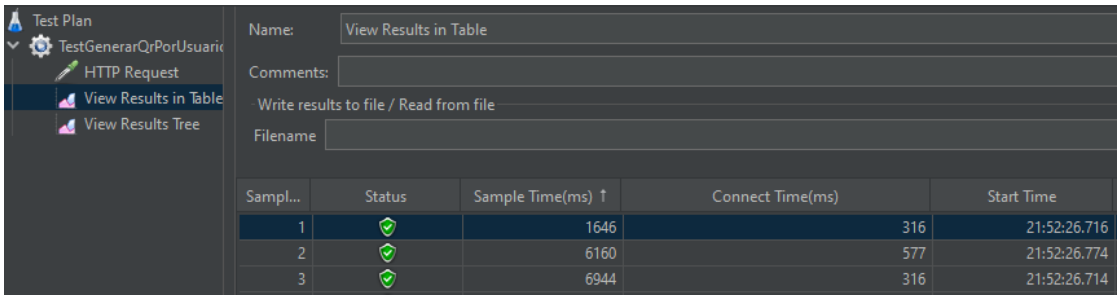

Figura 22

Carga concurrente QR 400 usuarios

A medida que se incrementa el número de peticiones, se observa un aumento en el tiempo promedio de respuesta. Por ejemplo, para 100 peticiones, el tiempo promedio fue de 2286ms, mientras que para 400 peticiones, este tiempo aumentó a 7810ms. Sin embargo, es notable que, a pesar del aumento de peticiones, la duración total no se incrementa de manera proporcional, lo cual es un indicativo favorable. Es importante analizar el tiempo máximo, que para 400 peticiones alcanzó los 8685ms, lo que puede representar puntos a considerar para futuras optimizaciones. Estos datos proporcionan información valiosa para garantizar que

el sistema mantenga un desempeño óptimo ante situaciones de alto tráfico recurrente.

- **Herramienta:** Jmeter
- **EndPoint:** "/api/AspNetCoreIdentity/TestRealizarTransferencia"

Tabla 29

Pruebas de carga transferencias con peticiones concurrentes

Usuarios	Duración	Tiempo Promedio	Tiempo Máximo	Tiempo Mínimo	Figura
100	1s	257ms	459ms	152ms	Figura 23
250	2s	1060ms	1770ms	134ms	Figura 24
400	5s	2423ms	4359ms	345ms	Figura 25

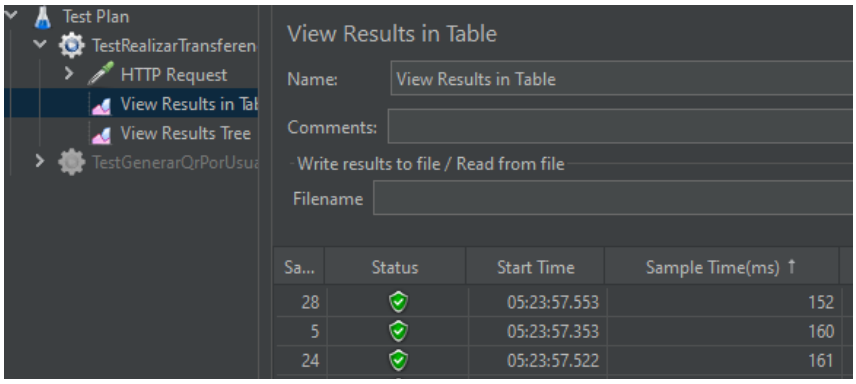

Figura 23

Carga concurrente transferencia 100 usuarios

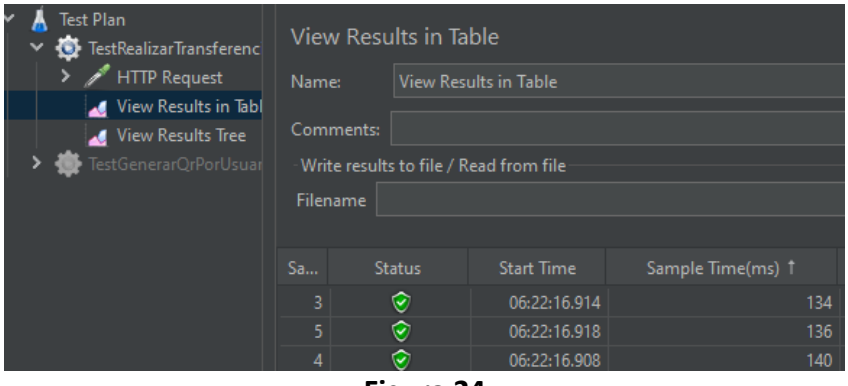

Figura 24

Carga concurrente transferencia 250 usuarios

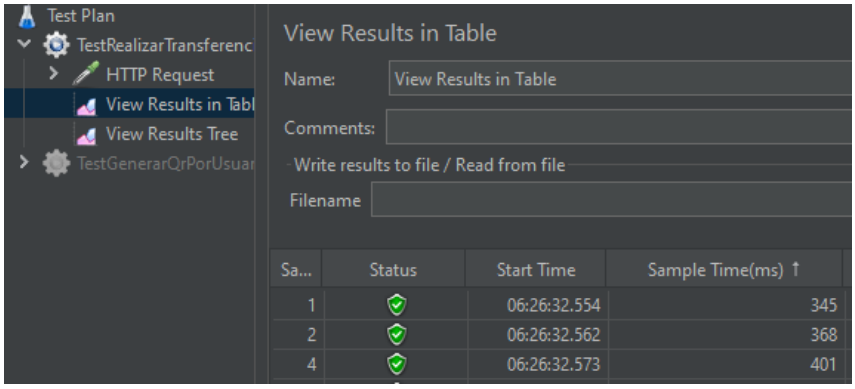

Figura 25

Carga concurrente transferencia 400 usuarios

Al evaluar los resultados, se aprecia que al aumentar el número de peticiones, el tiempo promedio de respuesta también incrementa. Específicamente, con 100 peticiones, el tiempo promedio se mantuvo en 257ms, pero al elevar las peticiones a 400, este valor se incrementó significativamente a 2423ms. De manera similar, el tiempo máximo también presentó un aumento considerable. Es esencial señalar que el tiempo mínimo, en general, se mantuvo relativamente bajo en todas las pruebas, lo que indica que algunas peticiones se procesaron con gran eficiencia.

Este tipo de pruebas son cruciales para entender cómo se comporta el sistema en operaciones específicas, como las transferencias, y así poder identificar áreas de optimización para mejorar el desempeño y garantizar una experiencia de usuario óptima, incluso bajo condiciones de alta demanda.

4.3.3 Pruebas de estrés

- **Herramienta:** Jmeter
- **EndPoint:** "/api/AspNetCoreIdentity/TestGenerarQrPorUsuario"

La resistencia de un sistema ante situaciones extremas es fundamental para garantizar su robustez y confiabilidad en circunstancias atípicas. Con la intención de determinar los límites del sistema y su comportamiento en estos escenarios, se llevaron a cabo pruebas de estrés utilizando la herramienta Jmeter

Tabla 30

Pruebas de estrés QR

Usuarios	Duración	Tiempo Promedio	Tiempo Máximo	Tiempo Mínimo	Fallas	Figura
500	12s	9271ms	11751ms	354ms	46	Figura 26
1000	9s	3759ms	9358ms	605ms	553	Figura 27

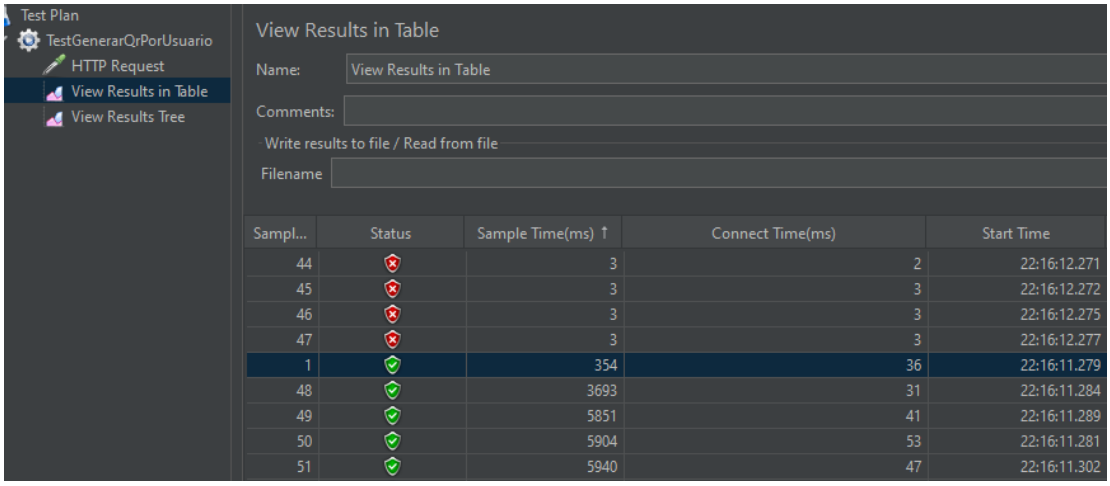

Figura 26

Prueba estrés QR 500 usuarios

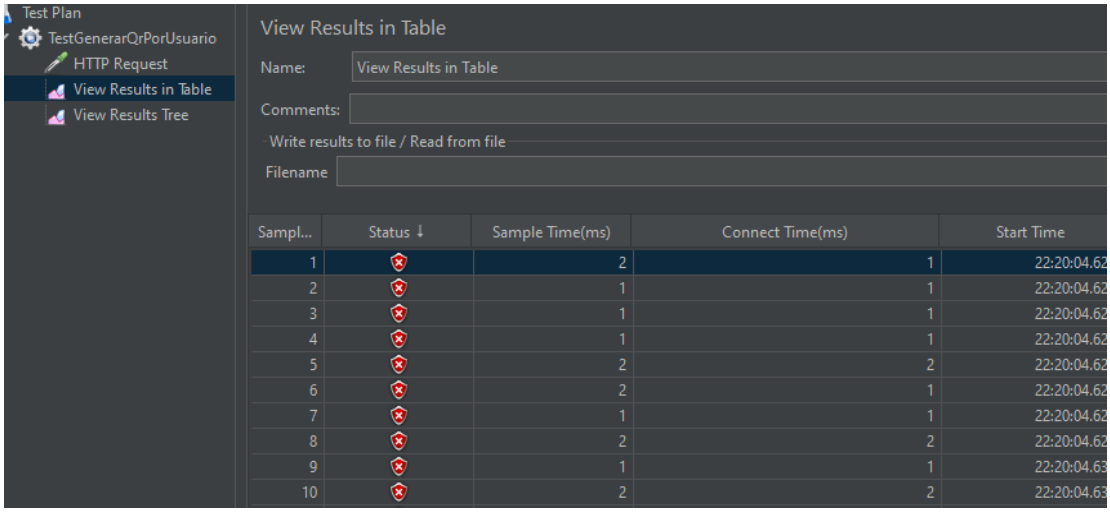

Figura 27

Prueba estrés QR 1000 usuarios

Es importante resaltar que, en las pruebas de estrés, más allá de los tiempos de respuesta, es esencial observar el número de fallas. En la prueba con 500 peticiones, se registraron 46 fallas, lo que indica ciertos fallos en la capacidad de respuesta del

sistema. Al duplicar las peticiones a 1000, las fallas se incrementaron significativamente, llegando a 553. Sorprendentemente, a pesar del aumento de fallas, la duración total fue menor y el tiempo promedio de respuesta se redujo, lo que podría sugerir una saturación del sistema, llevándolo a rechazar peticiones más rápidamente.

Estos resultados resaltan la importancia de realizar pruebas de estrés para identificar y corregir potenciales vulnerabilidades en el sistema antes de que enfrenten situaciones reales de alta carga, garantizando así un servicio más confiable y robusto para los usuarios.

- **Herramienta:** Jmeter
- **EndPoint:** "/api/AspNetCoreIdentity/TestRealizarTransferencia"

Para profundizar el entendimiento de la resistencia y confiabilidad del sistema, particularmente en las operaciones de transferencia, se ejecutaron pruebas de estrés. El propósito era observar cómo se comporta el sistema ante una avalancha de solicitudes de transferencias, que es una de las operaciones críticas en muchas aplicaciones financieras. Para estas pruebas, se empleó la herramienta Jmeter.

Los resultados obtenidos se presentan a continuación:

Tabla 31
Pruebas de estrés transferencia

Usuarios	Duración	Tiempo Promedio	Tiempo Máximo	Tiempo Mínimo	Fallas	Figura
500	6s	2960ms	5528ms	552ms	69	Figura 28
1000	5s	1554ms	5291ms	738ms	588	Figura 29

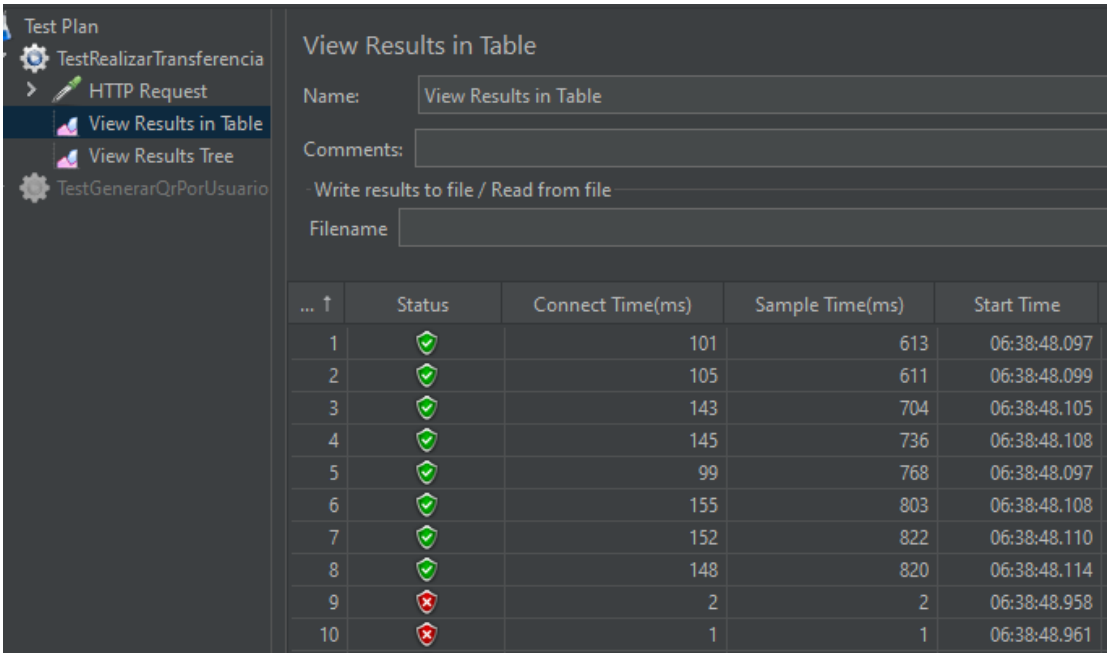

Figura 28

Prueba estrés transferencia 500 usuarios

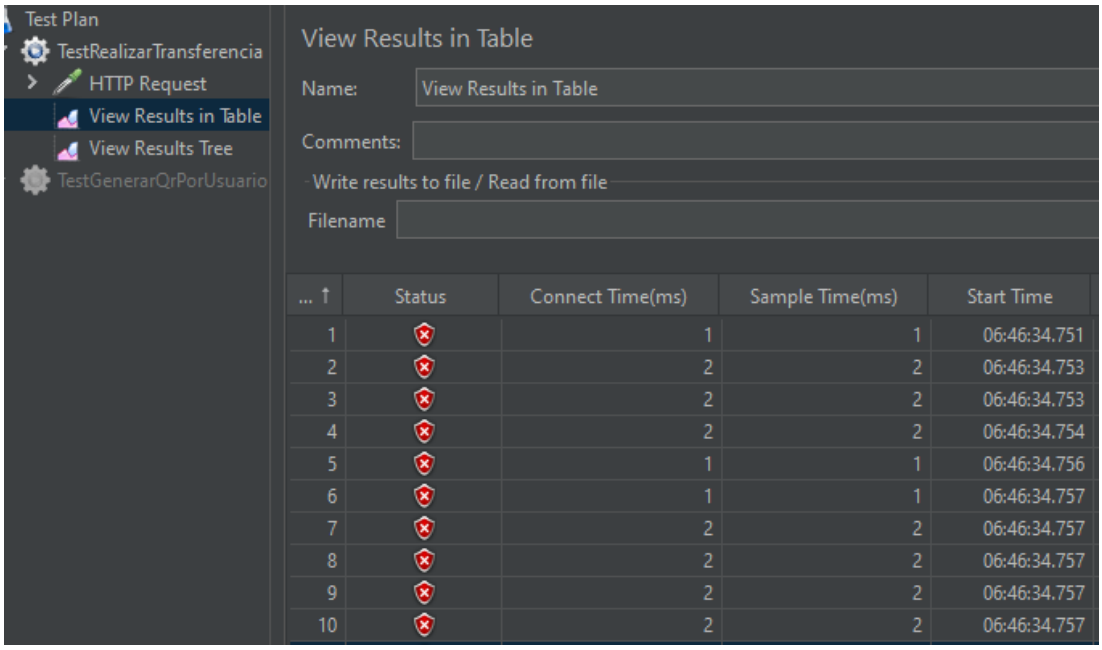

Figura 29

Prueba estrés transferencia 1000 usuarios

Al analizar los resultados, es notable que al duplicar el número de peticiones de 500 a 1000, las fallas aumentaron considerablemente, pasando de 69 a 588. A pesar de este aumento, la duración total fue ligeramente menor y el tiempo promedio de respuesta se redujo a la mitad. Este comportamiento sugiere que, al enfrentar una

mayor carga, el sistema puede haber empezado a rechazar rápidamente ciertas peticiones para no sobrecargarse, resultando en un mayor número de fallas, pero con tiempos de respuesta más rápidos.

Estos hallazgos recalcan la necesidad de fortalecer y optimizar el componente de transferencia, dada su relevancia y la alta cantidad de fallas observadas durante estas pruebas. Es crucial garantizar que, en situaciones reales de alta demanda, el sistema pueda manejar las solicitudes de transferencia de manera eficiente y confiable.

5. Conclusiones

a. El segundo factor de autenticación no solo es necesario para sistemas financieros transaccionales, sino que también actúa como una barrera contra el fraude y el robo de identidad, incrementando significativamente la seguridad más allá de las contraseñas tradicionales, que a menudo son vulnerables a ataques de fuerza bruta y tácticas de ingeniería social.

b. Al establecer la arquitectura para la generación de OTP, es crucial incorporar prácticas de seguridad robustas, como la encriptación de extremo a extremo y el almacenamiento seguro de claves, para salvaguardar la integridad y confidencialidad del proceso de autenticación en todos los componentes del entorno de la aplicación, desde el usuario hasta el backend.

c. La adopción generalizada de OTP en el sector financiero, ya sea a través de SMS, correo electrónico o aplicaciones móviles, refleja su eficacia como mecanismo de autenticación. Su capacidad para ofrecer un método seguro y fácilmente accesible para la verificación de identidad lo ha establecido como un estándar de facto en la industria para proteger las transacciones y la información sensible del cliente.

d. La implementación de métodos de verificación para las solicitudes que el servidor recibe es imprescindible para prevenir ataques y vulnerabilidades. Estos métodos deben incluir validaciones de la carga de la solicitud, autenticación del token de seguridad y el aseguramiento de que cada petición a la API cumpla con políticas de control de acceso, lo que proporciona una capa adicional de defensa contra el acceso no autorizado y las manipulaciones de datos.

e. La implementación de la autenticación multifactor, incluyendo OTPs, tiene un impacto significativo en la percepción de seguridad por parte de los usuarios, reforzando la confianza en las instituciones financieras. Al garantizar la seguridad de las cuentas y las transacciones, los usuarios se sienten más seguros

al realizar operaciones en línea, lo que a su vez puede aumentar la adopción y frecuencia de uso de servicios financieros digitales.

f. La flexibilidad inherente de la autenticación OTP permite su adaptabilidad a diversos entornos y plataformas tecnológicas, lo que facilita la escalabilidad de los sistemas financieros para acomodar un creciente número de usuarios y transacciones. Esta capacidad de escalado es esencial para mantener la seguridad sin comprometer la experiencia del usuario en un panorama digital en constante evolución.

g. A pesar de su amplia adopción, la autenticación basada en OTP enfrenta desafíos continuos en términos de seguridad, como la vulnerabilidad a ataques de interceptación y técnicas de phishing. La investigación y el desarrollo continuo en métodos de entrega más seguros y tecnologías de autenticación, como la biometría y la autenticación basada en la ubicación, serán cruciales para mantener la eficacia de los OTPs como una herramienta de seguridad en el futuro.

6. GLOSARIO

OTP: La contraseña de un solo uso denominada OTP por sus siglas en inglés (One Time Password). Proporciona un nivel adicional de seguridad, siendo muy común y útil para asegurar los accesos de sistemas transaccionales en línea.

IDE: El entorno de desarrollo integrado IDE por sus siglas en inglés (Integrated Development Environment) es una aplicación de software que embebe o se integra con herramientas como editor de código, gestor de versiones, compilador, depurador, entre otros. Pensado en la productividad del desarrollo de software haciendo que sea cómodo y eficiente para el desarrollador.

2FA: Segundo Factor de Autenticación o Two factor Authentication, se trata de una de las técnicas ampliamente utilizadas para prevenir accesos no autorizados a los servicios en línea. Como su nombre sugiere es un segundo factor para autorizar el acceso a un sistema después de utilizar usuario y contraseña.

JWT: (JSON Web Token) es un estándar para transmitir información de manera segura entre dos partes mediante una cadena de caracteres firmada digitalmente, utilizado comúnmente en autenticación y autorización en aplicaciones web.

Código QR: Un código QR (Quick Response Code) es un tipo de código de barras bidimensional que puede almacenar información de forma eficiente y es fácilmente escaneable por dispositivos con cámara, como teléfonos inteligentes.

Token: Es un código generado temporalmente, utilizado para autenticar una sesión o transacción.

Referencias

Krishna, S. P., Tejasri, D., Soumya, B., Madhuri, M., & Lubna. (2022). Bank Application: One-Time Password Generation. 2022 International Conference on Applied Artificial Intelligence and Computing (ICAAIC), 855-859. https://doi.org/10.1109/ICAAIC53929.2022.9792823

Naik, R., & Singh, U. (2021). Secured 6-Digit OTP Generation using B-Exponential Chaotic Map. International Journal of Advanced Computer Science and Applications, 12(12). https://doi.org/10.14569/IJACSA.2021.0121296

Kaur, N., Devgan, M., & Bhushan, S. (2016). Robust login authentication using time-based OTP through secure tunnel. In Proceedings of the 2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom) (pp. 3222–3226).

Imamah. (2018). One Time Password (OTP) Based on Advanced Encrypted Standard (AES) and Linear Congruential Generator(LCG). 2018 Electrical Power, Electronics, Communications, Controls and Informatics Seminar (EECCIS), 391-394. https://doi.org/10.1109/EECCIS.2018.8692931

Kansuwan, T., & Chomsiri, T. (2019). Authentication Model using the Bundled CAPTCHA OTP Instead of Traditional Password. 2019 Joint International Conference on Digital Arts, Media and Technology with ECTI Northern Section Conference on Electrical, Electronics, Computer and Telecommunications Engineering (ECTI DAMT-NCON), 5-8. https://doi.org/10.1109/ECTI-NCON.2019.8692255

Shyry, S. P., Mahithaasree, M., & Saranya, M. (2018). Implementation of One Time Password by 3 3 Vedic Multiplier. 2018 International Conference on Computer, Communication, and Signal Processing (ICCCSP), 1-5. https://doi.org/10.1109/ICCCSP.2018.8452861

Thomas, J., & Goudar, R. H. (2018). Multilevel Authentication using QR code based watermarking with mobile OTP and Hadamard transformation. 2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI), 2421-2425. https://doi.org/10.1109/ICACCI.2018.8554891

Janakiraman, S., Sree, K. S., Manasa, V. L., Rajagopalan, S., Thenmozhi, K., & Amirtharajan, R. (2018). OTP on Demand—An Embedded System for User Authentication. 2018 International Conference on Computer Communication and Informatics (ICCCI), 1-5. https://doi.org/10.1109/ICCCI.2018.8441400

Elias, E. P., Santhanavijayan, A., Janet, B., & Arul, K. R. J. (2022). OTP System Based on ECC Key Exchange. 2022 International Conference on Communication, Computing and Internet of Things (IC3IoT), 1-6. https://doi.org/10.1109/IC3IOT53935.2022.9768019

AlamgirKhan, A. (2013). Preventing Phishing Attacks using One Time Password and User Machine Identification. International Journal of Computer Applications, 68(3), 7-11. https://doi.org/10.5120/11557-6839

Islam Khan, B. U., Olanrewaju, R. F., Anwar, F., & Yaacob, M. (2018). Offline OTP based solution for secure internet banking access. In 2018 IEEE Conference on E-Learning, e-Management and e-Services (IC3e) (pp. 167–172). IEEE. https://doi.org/10.1109/IC3e.2018.8632643