# ESCUELA POLITÉCNICA NACIONAL

## FACULTAD DE INGENIERÍA DE SISTEMAS

## CREACIÓN DE UN PROTOTIPO DE SISTEMA DE ONE-TIME PASSWORD PARA UN SISTEMA DE AUTENTICACIÓN DE DOS FACTORES

**Implementación de un prototipo de sistema generador de OTP basado en TOTP y su envío mediante un aplicativo web**

**TRABAJO DE INTEGRACIÓN CURRICULAR PRESENTADO COMO REQUISITO PARA LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN CIENCIAS DE LA COMPUTACIÓN**

**BRAYAN ALEXIS FERNÁNDEZ GONZA**

**brayan.fernandez@epn.edu.ec**

**DIRECTOR: Sang Guun Yoo, PhD**

**sang.yoo@epn.edu.ec**

**DMQ, febrero 2023**

# CERTIFICACIONES

Yo, BRAYAN ALEXIS FERNÁNDEZ GONZA declaro que el trabajo de integración curricular aquí descrito es de mi autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen en este documento.

**BRAYAN ALEXIS FERNÁNDEZ GONZA**

Certifico que el presente trabajo de integración curricular fue desarrollado por BRAYAN ALEXIS FERNÁNDEZ GONZA, bajo mi supervisión.

**SANG GUUN YOO**
**DIRECTOR**

# DECLARACIÓN DE AUTORÍA

A través de la presente declaración, afirmamos que el trabajo de integración curricular aquí descrito, así como el (los) producto(s) resultante(s) del mismo, son públicos y estarán a disposición de la comunidad a través del repositorio institucional de la Escuela Politécnica Nacional; sin embargo, la titularidad de los derechos patrimoniales nos corresponde a los autores que hemos contribuido en el desarrollo del presente trabajo; observando para el efecto las disposiciones establecidas por el órgano competente en propiedad intelectual, la normativa interna y demás normas.

BRAYAN ALEXIS FERNÁNDEZ GONZA

SANG GUUN YOO, PhD.

LUIS ERNESTO ALMEIDA ZAMBRANO

DALIANA ZAMBRANO PEREDA

ANTHONY ISRAEL ALMACHI CHASI

HILTON BLADIMIR PILLAJO ANAGUANO

# DEDICATORIA

A mis amados padres, quienes son la razón de mi vida. Sin su amor y apoyo incondicional, este logro no habría sido posible. A mi hermano, que ha traído momentos de risas y felicidad a mi vida. A todas aquellas personas que me han apoyado durante este trayecto.

# AGRADECIMIENTO

A mis padres Clemencia y Germán, quienes han sido mi motivación e inspiración para no rendirme y lograr superar todos los obstáculos que se me han presentado. Los amo con todo mi corazón.

A mi hermano Mateo, quien con su cariño, amistad y locuras ha traído momentos de alegría a mi vida.

A mi tutor en este proyecto, Sang Guun Yoo, quien con su conocimiento y consejos me ha guiado y motivado en esta etapa académica.

A mis amigos incondicionales, Jordan, Javier y Rony quienes se han convertido en parte de mi familia y me han brindado momentos llenos de anécdotas y gratos recuerdos.

A mis compañeros de carrera, Luis, Daliana y Cristhian, quienes durante estos años me han brindado su apoyo y amistad.

Al ser lleno de sueños, determinación y perseverancia que se ha mantenido en mí y que me ha permitido no tener miedo de ir tras lo que quiero.

# ÍNDICE DE CONTENIDO

# ÍNDICE DE FIGURAS

# RESUMEN

La autenticación representa un proceso importante de seguridad en el control de acceso a una red o sistema. Cuando este proceso se basa en algo que uno sabe, se trata de la autenticación mediante un usuario y una contraseña. No se puede depender únicamente de las contraseñas tradicionales o estáticas, pues son vulnerables ante diversos tipos de ataques informáticos. Una solución común a este problema es la implementación de la autenticación mediante One-Time Password (OTP). En este proyecto, se propone la creación de un prototipo de OTP para la implementación de la autenticación de doble factor. Primero, se realiza una revisión de los distintos tipos de OTP en base a los algoritmos de generación, métodos de entrega y su nivel de seguridad. Luego, siguiendo una metodología de trabajo se realiza el desarrollo del componente para tener como resultado el prototipo para generación de OTP. Por último, se detallan los resultados obtenidos respecto a la generación de OTP y el funcionamiento del prototipo, las conclusiones y recomendaciones del proyecto.

**PALABRAS CLAVE:** One-Time Password, HOTP, Time-based OTP, Sistema de autenticación.

# ABSTRACT

Authentication represents an important security process in controlling access to a network or system. When this process is based on something one knows, it is referred to as authentication through a username and password. Traditional or static passwords cannot be solely relied upon as they are vulnerable to various types of cyber-attacks. A common solution to this problem is the implementation of authentication through One-Time Password (OTP). In this project, the creation of an OTP prototype for implementing two-factor authentication is proposed. First, a review of the different types of OTPs based on generation algorithms, delivery methods, and their level of security is performed. Then, following a working methodology, the component development is carried out to result in the OTP generation prototype. Finally, the results obtained regarding the generation of OTP and the operation of the prototype are detailed, as well as the conclusions and recommendations of the project.

**KEYWORDS:** One-Time Password, HOTP, Time-based OTP, authentication system.

# 1 INTRODUCCIÓN

## 1.1 Descripción del proyecto

La autenticación es un aspecto fundamental que tomar en cuenta cuando se trata de la seguridad en línea, y se ha vuelto aún más importante conforme más y más servicios migran al mundo digital. Las contraseñas tradicionales, basadas en un usuario y una contraseña, con el pasar del tiempo se han vuelto mucho más vulnerables ante ataques de piratas informáticos, robo de identidad y otros tipos de delitos cibernéticos. Por tal motivo, el uso de factores de autenticación como datos biométricos, tokens y OTPs (One-Time Passwords) son medidas adicionales de seguridad cada vez más comunes en todo el mundo [1].

Según [2], una OTP es una contraseña válida únicamente por un corto periodo de tiempo para la validación de una transacción o inicio de sesión. Muchas empresas y organizaciones hacen uso de las OTPs para asegurarse que únicamente los usuarios autorizados puedan acceder a sus servicios. La OTP puede ser generada por distintos algoritmos y entregada por distintos medios.

Con el propósito de conocer los diferentes algoritmos para la generación de OTPs y sus distintos métodos de entrega, se propone la creación de un prototipo de sistema de OTP para un sistema de doble factor de autenticación (2FA).

La arquitectura del proyecto general se muestra en la Figura 1, en dónde se tiene un servidor y una aplicación móvil. El servidor será el encargado de generar, enviar y validar las OTPs, mientras que la aplicación móvil permitirá el ingreso de la OTP por parte del usuario que, de hacerlo correctamente, será autenticado. Dentro de esta arquitectura general, los componentes a ser desarrollados en el presente trabajo son los servicios de generación, validación y envío del OTP.


**Figura 1**. Arquitectura del proyecto general.

## 1.2 Descripción del componente

El desarrollo de este componente esta dado por dos fases principales, la primera fue el análisis y selección de un tipo de One-Time Password (OTP) y, la siguiente, la creación de un prototipo de un sistema de OTP para un sistema de autenticación de dos factores. Entre las principales tareas desarrolladas para el presente componente se encuentran la generación de un OTP, su respectiva validación y su implementación para su visualización en un prototipo de aplicativo web.

La primera parte del componente inicia con el planteamiento a la pregunta de investigación, la cual fue formulada de la siguiente manera: ¿Qué es un OTP y cuáles son sus aplicaciones?, partiendo de esta pregunta, se realizó el estudio del arte, con el fin de conocer los diferentes tipos de OTP, los métodos de entrega y su nivel de seguridad. Con la realización de este estudio se procedió a establecer el tipo de OTP a ser implementado. Además, el estudio del arte permitió conocer los requerimientos para el desarrollo de la arquitectura del prototipo a ser implementado.

El tipo de OTP elegido a ser implementado fue el TOTP (Time-based One-Time Password). TOTP es una variante de OTP con una mayor seguridad debido a que el código cambia con el tiempo, lo que resulta en una mayor dificultad para los atacantes adivinar el código. El método de entrega del TOTP se realiza mediante una aplicación web en dónde se muestra el código generado, así como también el tiempo restante para que el código caduque.

Para la validación del OTP se ha implementado una API que permite el envío del OTP desde la aplicación móvil utilizando encriptación para una comunicación segura. En la Figura 2 se puede observar que el componente respectivo del presente trabajo se encuentra remarcado.

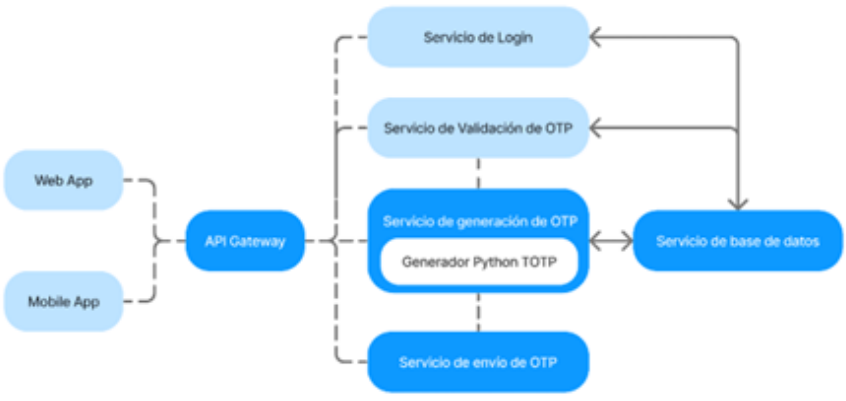



**Figura 2.** Arquitectura del componente desarrollado.

## 1.3 Objetivo general

Crear un prototipo de un sistema de autenticación One-Time Password (OTP) para un sistema de doble factor de autenticación mediante el uso del algoritmo Time-based One-Time Password (TOTP).

## 1.4 Objetivos específicos

1. Entender el estado del arte respecto a sistemas de doble factor de autenticación y métodos de generación de OTPs con el fin de identificar los diferentes tipos de OTP, sus métodos de entrega más comunes y sus niveles de seguridad.
2. Desarrollar un prototipo de sistema TOTP que tenga la capacidad de generar, enviar y validar TOTPs en tiempo real.
3. Implementar un aplicativo web que muestre el código TOTP generado por el prototipo de sistema OTP.

## 1.5 Alcance

El proyecto actual tiene como objetivo crear un prototipo de un sistema de OTP para un sistema de autenticación de dos factores, sin embargo, no se pretende que este sistema sea implementado en un entorno o sistema real.

# 2 MARCO TEÓRICO

En esta sección se tratan los conceptos más relevantes que fueron revisados para el estado del arte y utilizados para el desarrollo del componente propuesto.

## 2.1 OTP

Una One-Time Password (OTP) es una contraseña de un solo uso (contraseña dinámica) que se utiliza para autenticar a un usuario, transacción o proceso en un sistema. Fue propuesto inicialmente en 1980 por Lamport [3], quien propone un algoritmo matemático que genera una lista de valores que pueden ser utilizados una sola vez. Esta es una secuencia en dónde cada valor de la lista depende del valor anterior para su generación. También describe una fase de registro que se ejecuta una sola vez y una fase de autenticación que se ejecuta tantas veces como el usuario acceda al sistema [4].

La autenticación por OTP es ampliamente reconocida como un método altamente efectivo para proporcionar acceso a un usuario autorizado. Las contraseñas son generadas por parte del servidor y enviadas al usuario, que a su vez envía de vuelta la contraseña recibida y si corresponde con la generada en el servidor, entonces se finaliza de manera exitosa la autenticación. De esta manera, el riesgo de sufrir un ataque en donde las contraseñas sean interceptadas o robadas es muy bajo. Aún si un atacante lograra obtener la OTP, no representaría un peligro debido a que no sería capaz de adivinar la siguiente OTP [4]. Una de las mayores ventajas con las que cuenta OTP respecto a las contraseñas estáticas o tradicionales es que no es vulnerable ante ataques de replay [5].

## 2.2 Algoritmos de generación de OTP

Existen distintos algoritmos propuestos para la generación de One-Time Passwords. En [6] se presenta un nuevo modelo para la generación de OTP, el cual se basa en el uso de un multiplicador védico de 3*3. Como primer paso para la generación del OTP, se obtienen los detalles de inicio de sesión del cliente desde la Autoridad de Certificación, los cuales los convierte en 8 bits, y luego, también convierte la longitud del mensaje en 8 bits. Estos 16 bits resultantes son convertidos a decimal utilizando CB2D, lo que resulta en dos dígitos decimales de 3 cifras que se multiplican entre sí mediante el multiplicador védico de 3*3, obteniendo de esta manera el OTP.

En [7] se indica una solución para superar las limitaciones que supone hacer uso del OTP finito propuesto por Lamport. Resalta las debilidades que tiene el OTP de Lamport como es la longitud de la cadena de hash finita, que resulta en que la generación del OTP es también finita. Por lo tanto, se ha diseñado una nueva cadena de hash para la generación del OTP de forma infinita, sin la necesidad de un secreto compartido entre las dos partes. A diferencia de la cadena larga del OTP de Lamport, la cadena de hash propuesta consiste en múltiples cadenas de hash cortas.

En [8] se propone un esquema de autenticación eficiente de doble factor basado en bases de datos negativas (NDB). Con este esquema, se logra la función de cambios de contraseñas, y las propiedades de la forma incierta de las bases de datos negativas pueden reducir la frecuencia de actualización de datos. Además, el esquema es resistente a la mayoría de los ataques, incluidos los ataques de adivinación de contraseña y los ataques de intermediario como man-in-the-middle.

En [9] se propone un modelo para la generación de OTP bancario mediante el uso de tres entradas diferentes conocidas por el cliente que cambian aleatoriamente y un vector de inicialización (IV). El generador propuesto es implementado utilizando un dispositivo móvil

basado en Android, en dónde el OTP es cifrado con AES-256 y enviado al cliente mediante un canal HTTP seguro.

## 2.3 Algoritmo HOTP

HOTP (HMAC-based One Time Password) es un algoritmo de generación de contraseñas de un solo uso basado en HMAC (Hash-based message authentication code) propuesto en [10]. Fue desarrollado para mejorar la seguridad en la autenticación de doble factor y aún sigue siendo un estándar ampliamente utilizado. A diferencia de OTP que genera una contraseña basada en un valor aleatorio, HOTP utiliza una secuencia de números crecientes. HOTP hace uso del algoritmo HMAC-SHA-1[11].

La implementación de HOTP implica la generación de una clave secreta compartida entre el servidor de autenticación y el usuario. Esta clave es utilizada para la generación de OTPs basados en un contador que se incrementa después de cada uso. Dado que el proceso de generación de contraseñas se basa en HMAC, se garantiza que la contraseña es única y no se puede reproducir sin la clave secreta. En la Figura 3 se observa el esquema de funcionamiento del algoritmo HOTP.

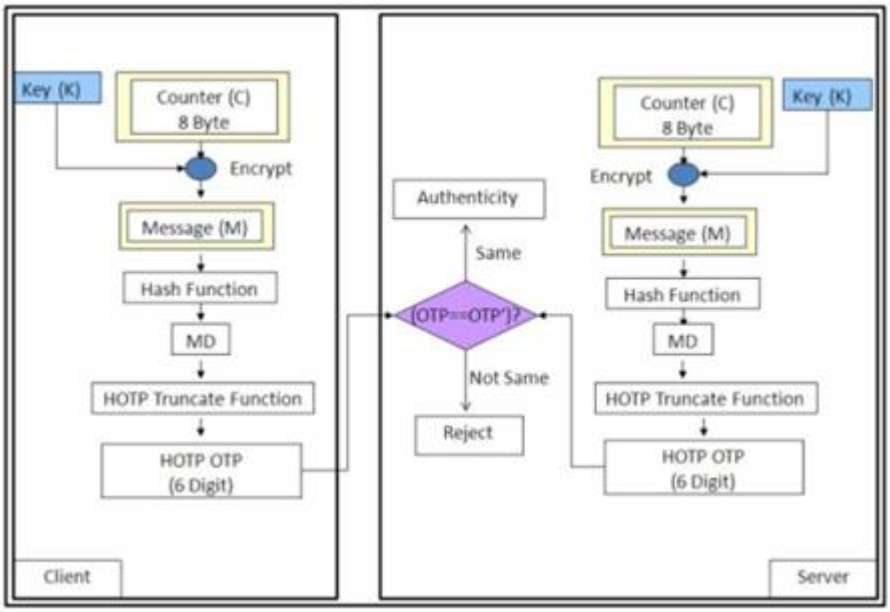



**Figura 3.** Esquema del algoritmo HOTP [12].

A pesar de estas medidas de seguridad, HOTP es vulnerable ante ataques de fuerza bruta debido al uso del valor truncado por HMAC-SHA-1. Esto supone implementar medidas adicionales de seguridad en el servidor de autenticación con el fin de prevenir este tipo de ataques.

## 2.4 Algoritmo TOTP

Según la RFC 6238 [13], TOTP es un mecanismo de autenticación que utiliza un código único y de un solo uso generado por un dispositivo móvil o una aplicación de software en función del tiempo actual y de una clave secreta compartida entre el servidor de autenticación y el dispositivo o la aplicación. TOTP es, además, una variante del algoritmo HMAC-based One-Time Password (HOTP) que en vez de generar el OTP mediante un contador, lo hace mediante el tiempo actual [14]. En la Figura 4 se puede observar el esquema de funcionamiento del algoritmo TOTP.

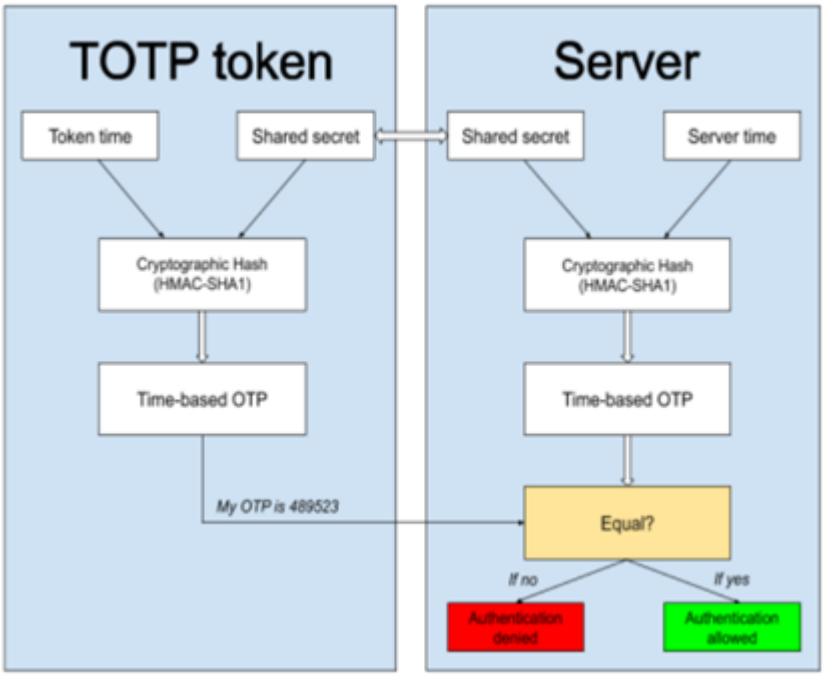



**Figura 4**. Esquema del algoritmo TOTP [12].

TOTP es ampliamente utilizado como un doble factor de autenticación para el acceso a diversos sistemas y servicios. TOTP utiliza el algoritmo HMAC-SHA-1 para la generación de la OTP, aunque también se puede hacer uso de otros algoritmos como HMAC-SHA-256 o HMAC-SHA512 para brindar una mayor seguridad [15]. Por lo general, el intervalo de tiempo para la generación de una nueva contraseña suele ser de 30 segundos, pero este valor puede ser ajustado de acuerdo con las necesidades del usuario o sistema. Para la generación del TOTP se genera una clave secreta que es compartida mediante un canal seguro entre el lado del cliente y el servidor. Se obtiene el tiempo actual y es dividido por el intervalo del tiempo para convertirlo a una cadena que será codificada mediante un algoritmo de encriptación como SHA-1 o SHA-256. Los últimos dígitos del hash resultante serán los valores OTP que, por lo general son 6 dígitos.

## 2.5 Métodos de entrega de OTP

### Mensajes de texto

El uso de mensajes de texto para la entrega de OTP resulta fácil de usar debido a que no se requiere ningún hardware o software especial. Además, la amplia adopción de los teléfonos móviles hace que este método de entrega sea accesible para una amplia gama de usuarios [1]. El código OTP llega a través de un mensaje de texto al teléfono móvil, en dónde el tiempo de caducidad del código generalmente es de 5 a 10 minutos.

Aunque el método de entrega de OTP mediante mensajes de texto resulta conveniente y sencillo de usar, también presenta ciertas desventajas y limitaciones. Debido a que la recepción de los mensajes de texto depende de la disponibilidad de la red móvil, estos pueden tardar en llegar debido a retrasos por en la entrega del proveedor o la falta de cobertura si se encuentra en zonas remotas. Además, también se tiene el riesgo de que, si un usuario llega a perder o dañar su teléfono móvil, no podrá recibir los códigos OTP y puede encontrarse con algunos problemas [16].

### Aplicaciones de autenticación

El método de entrega de OTPs mediante aplicaciones de autenticación es un proceso mediante el cual un código OTP es generado en tiempo real por una aplicación instalada en un dispositivo móvil o en una computadora y se utiliza para autenticar a un usuario. Entre algunos de los beneficios que se tienen al usar aplicaciones de autenticación se encuentra una mayor seguridad debido a que los códigos generados son únicos y en tiempo real, reduciendo de esta manera los ataques de phishing o de reutilización de contraseñas [17].

### Tokens propietarios

Los tokens propietarios son un método de autenticación utilizado en dispositivos físicos pequeños, como pueden ser llaves de hardware con una pequeña pantalla en dónde se muestra el código OTP. Tienen un amplio uso en tarjetas de control de acceso y tarjetas bancarias. Cómo se detalla en [18], si bien para su funcionamiento no es necesario que el usuario ingrese una contraseña o copie algún código enviado a una aplicación, puede ser necesario estar conectado a una conexión física o inalámbrica para el proceso de autenticación.

Sin embargo, entre algunas de sus desventajas se tiene que, debido a su tamaño es probable que se lleguen a perder o ser clonados. Debido a que requiere de un hardware

específico para su funcionamiento, pueden llegar a representar un alto costo de inversión y requieren un mantenimiento regular [19].

Por lo tanto, este método de entrega de OTPs puede ser adecuado para grandes organizaciones en dónde se requiere un alto nivel de seguridad y tengan la capacidad financiera para estar dispuestas a invertir en tokens de autenticación personalizados.

### Correo electrónico

El método de entrega de OTPs mediante correo electrónico es uno de los más comunes debido a su facilidad de uso, compatibilidad y rentabilidad. Al ser enviados directamente a través del correo electrónico, se reduce la carga de trabajo del usuario. La gran mayoría de servicios web admiten la autenticación de dos factores mediante correo electrónico y no tiene ningún costo ni requiere la compra de un dispositivo de hardware adicional. Sin embargo, una gran desventaja de este método de entrega es que los correos electrónicos pueden ser interceptados o comprometidos, poniendo en graves problemas de seguridad la autenticación de doble factor [20].

# 3 METODOLOGÍA

En esta sección se detalla la metodología seleccionada para el desarrollo del componente, así como las distintas fases que se realizaron. La metodología comenzó con la fase de investigación y planificación del proyecto, y a continuación con la ejecución y el desarrollo del respectivo componente. Con el propósito de lograr cumplir los objetivos planteados para el desarrollo del componente, la metodología seleccionada fue de desarrollo iterativo e incremental. Esta metodología es un conjunto de principios que facilita la administración de proyectos y permite su desarrollo gradual e iterativo hasta lograr el resultado deseado [21]. Entre algunas ventajas que aportó la metodología iterativa e incremental al proyecto, se encuentran las siguientes:

1. El número de iteraciones acordado con el equipo para el desarrollo del componente fue de cuatro iteraciones, con una duración de tres semanas cada una, con el propósito de tener mayor flexibilidad conforme el proyecto avanzaba y adaptar los cambios que sean necesarios.

2. Se tuvo reuniones semanales periódicas con el equipo para informar acerca de los avances realizados, asignación de tareas y con la ayuda de todos los miembros del equipo, abordar problemas y obstáculos que impidan el avance del proyecto. Esto permitió tener una mayor retroalimentación después de cada iteración, proporcionar

comentarios útiles y constructivos para un avance eficiente y adecuado del proyecto.

En la Figura 5 se muestra las distintas fases de la metodología iterativa-incremental llevadas a cabo durante el proyecto.

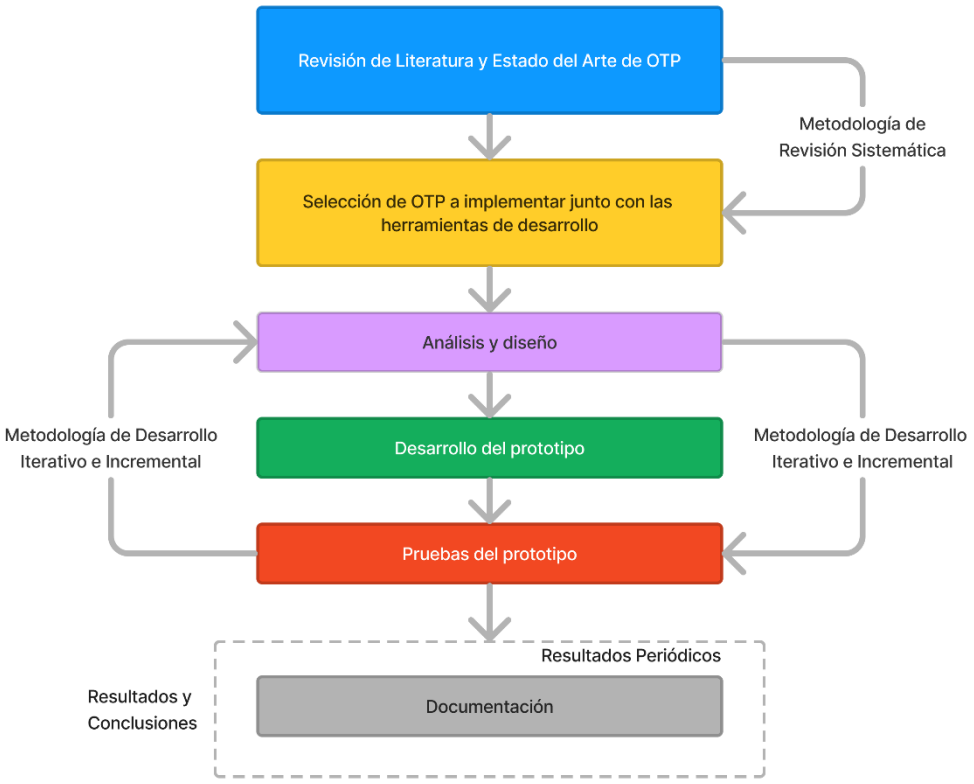



**Figura 5.** Metodología iterativa-incremental y las fases del proyecto.

Para la primera y segunda fase correspondiente a la revisión literaria y selección de herramientas, se utilizó la metodología de revisión sistemática (RS). Una RS es una metodología rigurosa y transparente que tiene como base una búsqueda exhaustiva y sistemática de la literatura para identificar los estudios relevantes. A partir de una evidencia disponible, las RS son capaces de brindar una síntesis crítica para dar con una respuesta frente a una pregunta planteada [22].

La primera etapa del proyecto consistió en una exhaustiva revisión literaria y estado del arte acerca de OTP, enfocado en los distintos tipos de OTP, los algoritmos de generación y sus distintas aplicaciones y soluciones en varios campos de la seguridad informática. Se definieron distintas cadenas de búsqueda que de acuerdo a los resultados obtenidos fueron reajustadas y la información obtenida fue clasificada. Scopus, IEEE Xplore y Elsevier fueron algunas de las bases de datos científicas en línea de las cuales se obtuvo la

información necesaria para cumplir con esta etapa del proyecto. Las distintas fases de la metodología de revisión sistemática de esta primera etapa se las puede observar en la Figura 6.

Para la segunda etapa se realizó una búsqueda de distintas herramientas para el respectivo desarrollo del componente y su prototipo. La arquitectura propuesta para el proyecto general se desarrolló tomando como referencia [23], y para el desarrollo del respectivo componente se toma como referencia a [13].

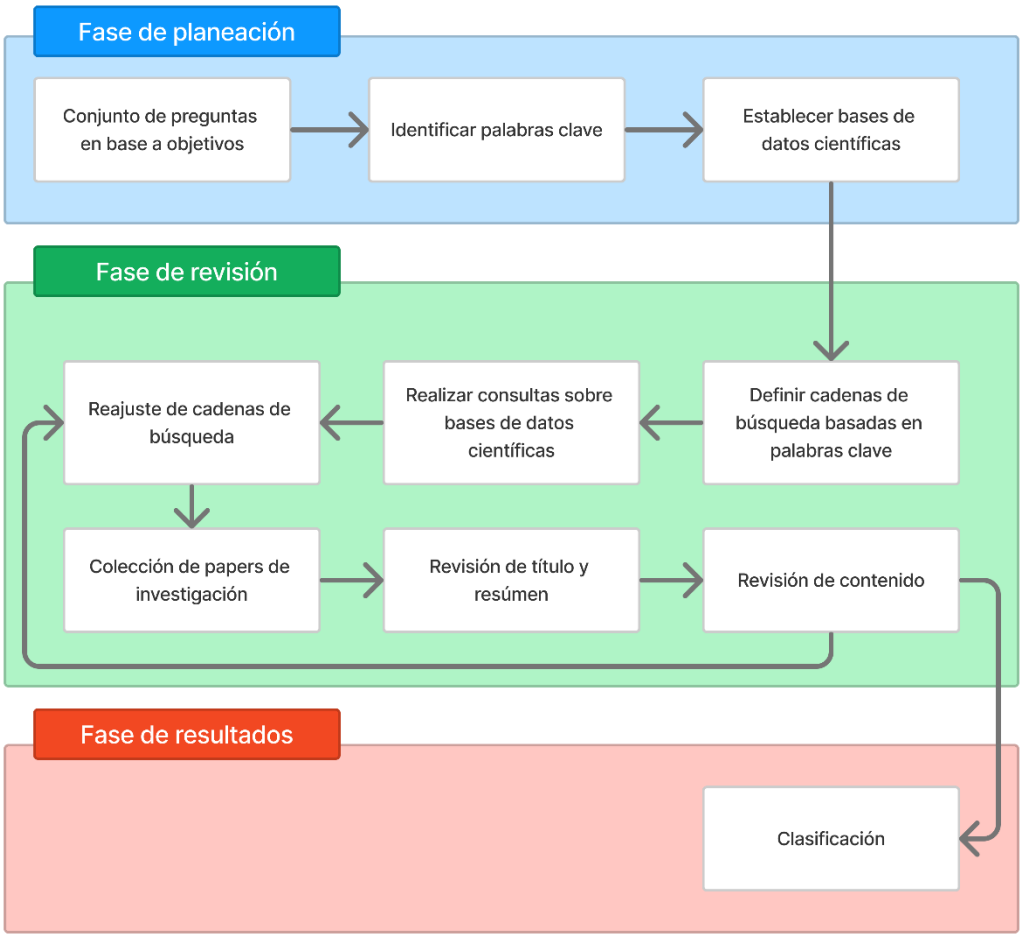



**Figura 6.** Metodología de revisión sistemática y sus fases.

Para una adecuada gestión de las referencias se hizo uso de Mendeley. Mendeley es un software para la gestión de referencias bibliográficas, que permitió trabajar en colaboración con los demás miembros del equipo con múltiples documentos y de esta forma, llevar un control adecuado de la bibliografía leída.

A partir de la tercera etapa se trabajó bajo la metodología iterativa-incremental en dónde cada incremento consta de 4 fases que son: análisis, diseño, desarrollo y pruebas.

En la fase de análisis se establecen las características operacionales y requisitos necesarios para el prototipo. Luego, en la fase de diseño se establece la solución a los requisitos previos y su implementación. En la fase de desarrollo se implementa la solución propuesta en el paso anterior. Por último, en la fase de pruebas se evalúa la solución implementada y en el caso de encontrarse errores, deberán ser solucionados en el siguiente incremento. En la Figura 7 se muestra el orden de cada incremento y sus fases.

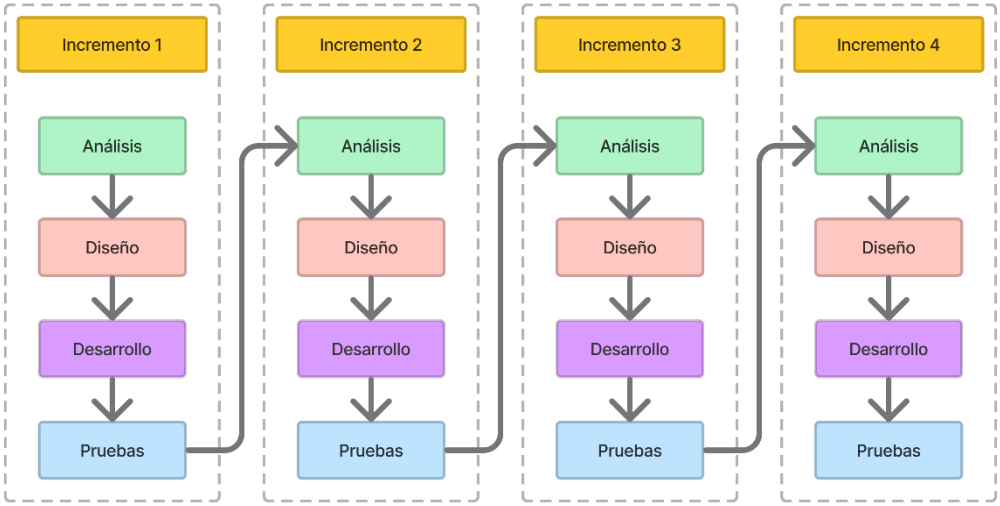



**Figura 7.** Incrementos de la metodología iterativa-incremental.

## 3.1 Selección de herramientas de desarrollo

### Python

Python es un lenguaje de programación de alto nivel que se caracteriza por su sintaxis simple y de fácil aprendizaje que se ha vuelto bastante popular en los años recientes. Debido a su versatilidad se lo utiliza en una amplia variedad de aplicaciones, que van desde desarrollo web hasta inteligencia artificial y machine learning (aprendizaje automático). Debido a su fácil integración con otros lenguajes y tecnologías, representa una excelente opción para proyectos en los que es necesario comunicarse con otros sistemas y dispositivos.

Además, posee una amplia variedad de librerías y herramientas disponibles que hacen sea más fácil y rápido el trabajo para los desarrolladores a la hora de implementar funcionalidades complejas [24]. Debido a que un gran porcentaje de proyectos relacionados con autenticación de doble factor han sido desarrollados en Python, este lenguaje de programación ha sido seleccionado por los beneficios descritos anteriormente.

### Flask

Para el desarrollo del backend de la aplicación se ha hecho uso del framework Flask. Este framework de desarrollo minimalista para Python fue creado por Ronacher en 2010 y actualmente es bastante popular en la comunidad de desarrolladores de Python debido a su simplicidad y flexibilidad [25]. Puede ser clasificado como un micro framework debido a que posee una base datos incorporada y no necesita ninguna otra herramienta o librería en particular. Además, existen extensiones para el framework como pueden ser de manejo de carga, validación de formularios y diferentes tecnologías de autenticación [26].

### PostgreSQL

El sistema para la gestión de base de datos seleccionado fue PostgreSQL. Actualmente es una de las bases de datos más avanzadas, confiables y de alta escalabilidad. Posee una alta seguridad al proporcionar múltiples capas de protección para los datos y también utiliza autenticación basada en roles y permisos de acceso para un control adecuado a los datos. Es altamente escalable y puede gestionar grandes cantidades de usuarios y datos simultáneamente [27]. Debido a que PostgreSQL incluye soporte para lenguajes de programación como Perl, Java y Python, además de los beneficios descritos anteriormente, fue el sistema de gestión de bases de datos (DBMS) seleccionado para el presente proyecto.

# 4 DESARROLLO DE LA SOLUCIÓN

## 4.1 Generación de OTP

### Análisis para la generación de OTPs

Para este primer incremento del proyecto, se realiza un análisis de los distintos generadores de OTP disponibles, así como los aspectos de seguridad a tomar en cuenta al momento de su implementación que se detallan en el Anexo I.

Además, con el fin de identificar las características clave que el prototipo debe tener para garantizar la seguridad de las cuentas de los usuarios se han creado sus respectivas historias de usuario. Las historias de usuario son una valiosa herramienta en un proyecto de desarrollo de software dado que permiten definir las necesidades y expectativas del usuario de manera clara y concisa. Cada historia de usuario proporciona un enfoque específico que se centra en los requisitos del usuario y establece un marco claro para el equipo en términos de qué características se deben implementar y cómo debe ser su funcionamiento. Las historias de usuario desarrolladas se presentan en el Anexo II.

### Diseño de arquitectura basada en generadores de OTP

Para la generación de los valores de OTP se analizado la librería PyOTP de Python. PyOTP puede ser usada para la implementación de un sistema de autenticación de doble factor (2FA) o multifactor (MFA) en aplicaciones web y móviles. PyOTP sigue los estándares definidos en los RFC 4226 [10] y RFC [13] en cuanto a los aspectos de seguridad para un despliegue seguro del algoritmo.

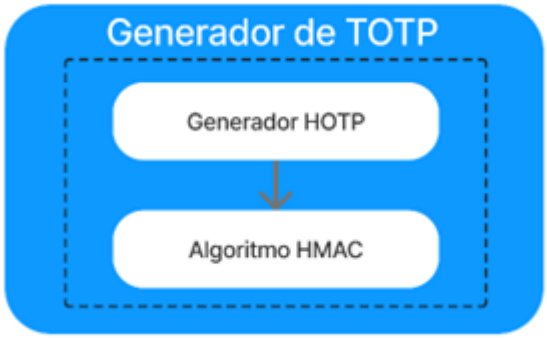



**Figura 8.** Generación de TOTP a partir de HOTP.

En la Figura 8 se detalla el funcionamiento del generador de OTP utilizando la función de TOTP. En este caso, el core de generación de OTP se basa en el algoritmo HMAC, pero que, en vez de hacer uso de un contador, ha sido adaptado una marca de tiempo para la generación del TOTP.

### Desarrollo de algoritmos basada en OTP

Una vez que han sido definidas las condiciones para la generación de OTP basados en tiempo, se procede a la implementación de la librería PyOTP. En la Figura 9 se detalla el código de generación de un OTP haciendo uso de un contador o marca de tiempo con el algoritmo HMAC.

```
def generate_otp(self, input: int) -> str:

        if input < 0:
            raise ValueError("input debe sr un entero positivo")
        hasher = hmac.new(self.byte_secret(), self.int_to_bytestring(input), self.digest)
        hmac_hash = bytearray(hasher.digest())
        offset = hmac_hash[-1] & 0xF
        code = (
            (hmac_hash[offset] & 0x7F) << 24
            | (hmac_hash[offset + 1] & 0xFF) << 16
            | (hmac_hash[offset + 2] & 0xFF) << 8
            | (hmac_hash[offset + 3] & 0xFF)
        )
        str_code = str(10_000_000_000 + (code % 10**self.digits))
        return str_code[-self.digits :]
```

**Figura 9.** Generación de OTP con el algoritmo HMAC.

Los códigos OTP necesitan de una semilla o clave secreta de 32 caracteres para su generación que se la puede generar de manera aleatoria mediante la función base32. Esta clave secreta resulta ser compatible con aplicaciones como Google Authenticator, WebAuthn o PyWARP y que, de ser necesario, pueden ser formateadas como cadenas hexadecimales codificadas.

### Pruebas de generación de algoritmos TOTP

El algoritmo de generación de OTPs es capaz de generar millones de diferentes posibilidades de combinaciones, pero surge un problema debido a que, por lo general, únicamente se muestran de 4 a 6 dígitos generados. Esto se discutirá en la sección de resultados. Por otra parte, es importante tomar en cuenta la resistencia proporcionada por cada algoritmo y protocolo frente a distintos tipos de ataques que se detalla en un análisis comparativo en el Anexo III.

## 4.2 Propuesta de arquitectura de componente e implementación de API

### Análisis y planificación de API basada en Flask

Para este segundo incremento del proyecto, se plantea la configuración e implementación de una API desarrollada en Flask para la comunicación entre los distintos módulos del prototipo y la generación de códigos OTP.

### Diseño para la implementación de API basada en Flask

En la Figura 10 se puede observar el diseño propuesto para la arquitectura de comunicación entre el generador de OTP y los demás módulos del prototipo a través de una API desarrollada en Flask.

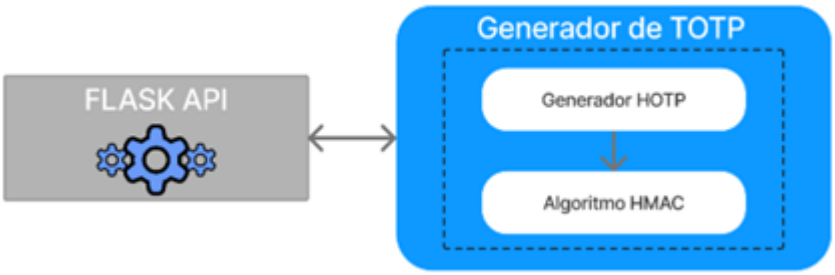



**Figura 10.** API Flask y conexión al generador de OTP.

La principal diferencia entre HOTP y TOTP es el mecanismo utilizado para generar la contraseña de un solo uso. Mientras que HOTP utiliza una clave secreta compartida y un contador, TOTP utiliza una clave secreta compartida y la hora actual del sistema para generar la OTP. Algunas aplicaciones esperan que la OTP generada sea de seis dígitos, pero este número puede variar de acuerdo con los requisitos del sistema.

### Desarrollo de la implementación de API Flask

Para el proceso de desarrollo de la API con Flask, es necesario tener instalado en el sistema el framework Flask para levantar un servidor e implementar la autenticación de OTP.

### Pruebas de la implementación de API Flask

Para el presente incremento, se llevaron a cabo pruebas destinadas a verificar una correcta conexión y envió de OTP entre un cliente mediante la API desarrollada en Flask y el generador de OTP.

## 4.3 Implementación y validación de OTPs

### Análisis para la validación de OTP

Para el tercer incremento del proyecto se realiza el análisis de la validación de un OTP y su respectiva implementación mediante el framework Flask. La API que ha sido generada en el incremento anterior será consumida por el aplicativo móvil en el lado del cliente y también por el aplicativo web para la visualización del OTP en tiempo real.

### Diseño de arquitectura para validación de OTP

La API desarrollada permite la comunicación entre el lado del cliente y el prototipo del sistema de autenticación con el propósito de recibir de manera cifrada los valores del código OTP por parte del cliente y enviarlos para su respectiva validación. En el caso en que el código OTP recibido coincida con el generado por el sistema, se enviará mediante la API al cliente indicando que el proceso de validación fue exitoso y permitiendo el acceso al sistema. En la Figura 11 se muestra la arquitectura propuesta para la validación de OTP.

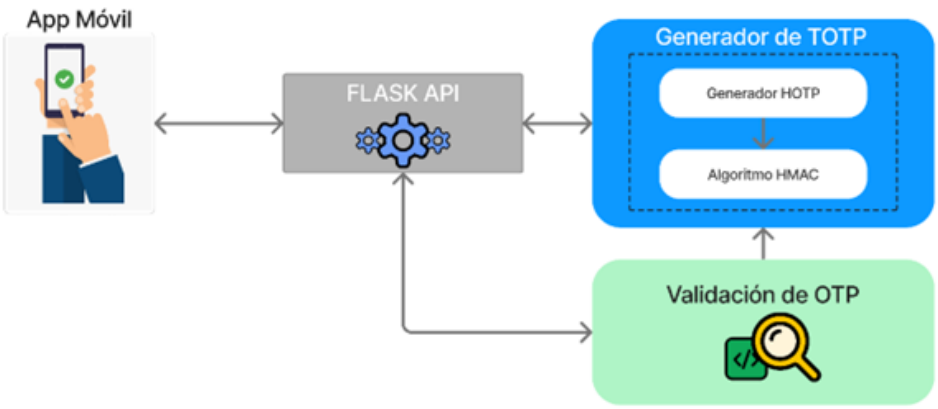



**Figura 11.** Validación de OTP mediante API Flask.

### Desarrollo de arquitectura para validación de OTP

El desarrollo del presente incremento comienza con la implementación y configuración de los endpoints en la API y que serán consumidos por la aplicación móvil para la recepción de los valores OTP ingresados por el usuario, así como el envío respectivo de la respuesta de la validación del OTP.

Una vez que el código OTP ha sido recibido, se procede a verificar que coincida con el valor generado por el algoritmo mediante la función de verificación que se muestra en la Figura 12. Si el valor OTP recibido coincide y, además, se encuentra dentro del tiempo establecido para su ingreso por parte del cliente, la validación será exitosa, caso contrario será fallida.

```
def verify(self, otp: str, for_time: Optional[datetime.datetime] = None,
valid_window: int = 0) -> bool:

        if for_time is None:
            for_time = datetime.datetime.now()

        if valid_window:
            for i in range(-valid_window, valid_window + 1):
                if utils.strings_equal(str(otp), str(self.at(for_time, i))):
                    return True
            return False

        return utils.strings_equal(str(otp), str(self.at(for_time)))
```

**Figura 12.** Código de validación de OTP.

### Prueba de validación de OTP

Para las pruebas de validación del OTP ingresado por el usuario, se realizaron distintos tipos de pruebas en dónde se variaban la generación de parámetros como la clave secreta, valores de tiempo, así como también los mismos valores de OTP con el fin de diagnosticar errores tanto en la misma generación del OTP como en la comunicación con la API y los valores enviados por el cliente.

```
>>> print(totp.now())
348436
>>> totp.verify(348436)
True
>>> totp.verify(123456)
False
```

**Figura 13.** Validación de TOTPs.

En la Figura 13 se muestra una prueba sencilla en donde se genera un valor TOTP, se ingresa un valor correcto y otro incorrecto y se tiene como salida un valor booleano de True y False, respectivamente.

## 4.4 Envío de TOTPs y prototipo de aplicativo web

### Análisis para el envío de OTP al aplicativo web y su visualización en tiempo real

Para el cuarto incremento del proyecto, se establece la conexión con el aplicativo web para la visualización en tiempo real del TOTP. Los valores generados por el algoritmo TOTP deberán ser validados de acuerdo con un listado de usuarios a los que se consultará en el proceso de validación de los TOTP.

### Diseño del envío de OTP al aplicativo web y su visualización en tiempo real

Para la respectiva validación del OTP se propone la arquitectura que se muestra en la Figura 14. La conexión con el aplicativo web se lo realiza mediante la API de Flask implementada en los incrementos anteriores. El servicio de gestión de base de datos ha

sido implementado en PostgreSQL mediante el uso de funciones hash para el cifrado de las contraseñas almacenadas.

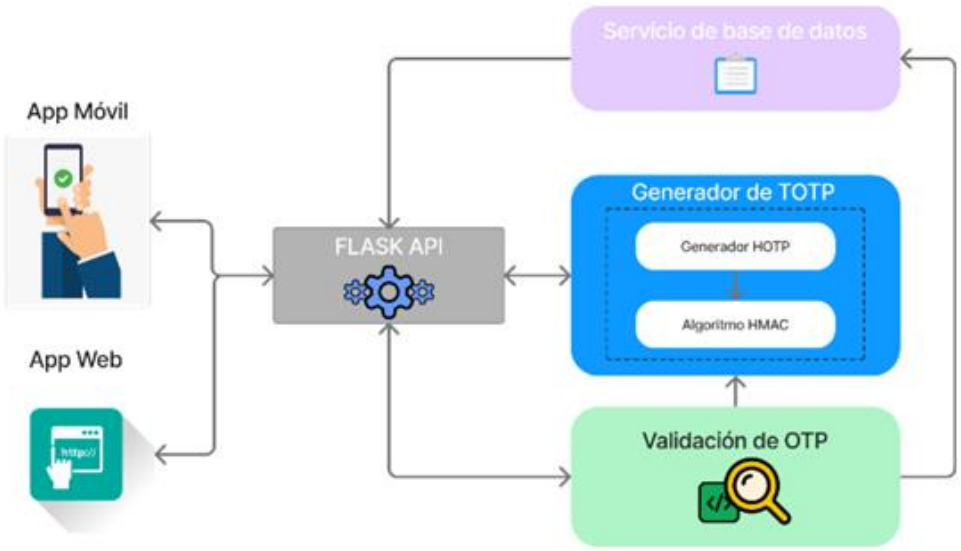



**Figura 14**. Validación de OTP mediante API de Flask.

Para el respectivo desarrollo del aplicativo web se ha creado el mockup de alto nivel. El uso de mockups en el desarrollo del prototipo de sistema OTP aporta varios beneficios como una visualización previa del diseño de la interfaz antes de empezar con la implementación del sistema, una retroalimentación temprana por parte de los usuarios y el resto de miembros del equipo facilitando la comunicación y discutiendo los posibles problemas y soluciones. Los mockups desarrollados para el presente proyecto se detallan en el Anexo IV.

### Desarrollo del envío de OTP al aplicativo web y su visualización en tiempo real

El aplicativo web fue desarrollado a partir de la extensión Flask-Bootstrap. Se ha implementado un inicio de sesión en dónde el usuario debe ingresar un nombre de usuario que puede ser un correo electrónico y una contraseña. Si esta información coincide con la información almacenada en la base de datos, entonces al usuario se le mostrará la página con los valores OTP que deberá ingresar en el aplicativo web para su autenticación. Para fines de seguridad la contraseña es guardada en forma cifrada en la base de datos para evitar que un atacante pueda ingresar al sistema en caso de romper la seguridad de la base de datos. La pantalla de ingreso se detalla en el Anexo V y la visualización del OTP en el Anexo VI. El código del proyecto se encuentra en el Anexo VIII.

Además, se han realizado evaluaciones de usabilidad para el aplicativo web desarrollado. Las pruebas de usabilidad son un conjunto de técnicas y métodos que se utilizan para evaluar la facilidad de uso de un producto o sistema por parte de los usuarios. Estas pruebas se realizan con el fin de identificar los problemas y obstáculos que pueden enfrentar los usuarios al interactuar con un producto o sistema. En la Tabla 1, se describen las preguntas realizadas a 16 personas en base a las heurísticas de Nielsen.

**Tabla 1.** Heurísticas de Nielsen.

| Heurística | Pregunta |
|---|---|
| H1 | ¿El sistema mantiene informados a los usuarios sobre su progreso? |
| H2 | ¿El sistema habla el idioma de los usuarios, con palabras, frases y conceptos familiares para el usuario, en lugar de términos orientados al sistema? |
| H3 | ¿El sistema muestra acciones que permiten al usuario repetir, salir o retroceder una acción previamente realizada? |
| H4 | ¿El sistema se maneja por un conjunto de estándares? |
| H5 | ¿El sistema presenta un diseño simple y fácil de usar que ayuda a que los usuarios tengan menos incidencia a cometer errores? |
| H6 | ¿El sistema presenta opciones, objetos y acciones que siempre están visibles para los usuarios, para que estos no tengan que recordarlas entre las distintas secciones del sistema? |
| H7 | ¿El sistema presenta aceleradores que permiten facilitar la interacción con el usuario? |
| H8 | ¿La interfaz del sistema es sencilla y fácil de entender? |
| H9 | ¿ Los mensajes de error se expresan en un lenguaje sencillo (sin códigos de error), indican con precisión el problema y sugieren una solución de manera constructiva ? |
| H10 | ¿El sistema brinda ayuda adecuada y precisa, como: instrucciones, videos de ayuda, guías, ¿entre otros ? |

#### Pruebas de envío de OTP al aplicativo web y su visualización en tiempo real

Para una correcta comunicación entre el aplicativo web y los servicios de generación y validación de OTP, se realizaron pruebas de funcionamiento con el uso de herramientas como Postman, para validar el correcto funcionamiento de las rutas de conexión el aplicativo web y móvil.

# 5 RESULTADOS, CONCLUSIONES Y RECOMENDACIONES

## 5.1 Resultados

La etapa de análisis y revisión de literatura respecto a OTP fue un aspecto fundamental para la comprensión de los distintos algoritmos existentes para la generación de OTPs. Como resultado de este análisis se logra definir el algoritmo de generación a

implementarse, su método de entrega y la arquitectura de desarrollo del proyecto. Un aspecto que se tomó en consideración fue el índice de repetición de códigos OTP de acuerdo con sus algoritmos de generación.

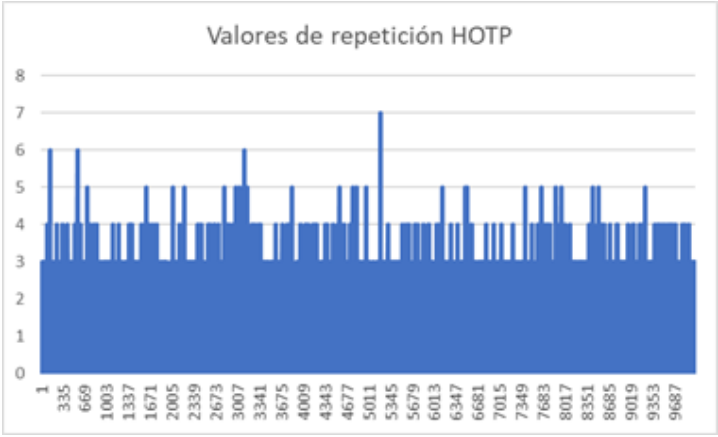



**Figura 15.** Tendencia de repetición de valores HOTP.

Mediante una misma llave secreta y un mismo número de ejecución de los algoritmos, se obtuvo los valores de repetición de códigos HOTP y TOTP. En la figura 15 se muestran los resultados obtenidos de generar 9999 valores de OTP mediante la utilización del algoritmo HOTP.

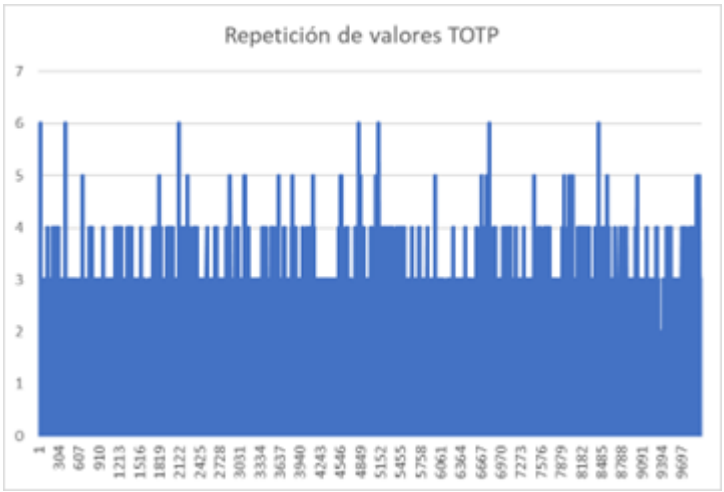



**Figura 16.** Tendencia de repetición de valores TOTP.

Existen valores que llegaron a repetirse hasta siete veces, mientras que otros no llegaron nunca a generarse. Un solo código generado por HOTP llegó a repetirse siete veces, lo que representa el 0,001% de todos los códigos generados. Por otra parte, 3711 códigos se generaron una sola vez, lo que representa el 37,11% de todos los códigos generados.

En la figura 16 se muestran los resultados obtenidos de realizar el mismo procedimiento para el algoritmo de generación de TOTP. A diferencia de HOTP, el valor más alto de

repetición fue de seis, pero con una mayor frecuencia que como sucedió con HOTP. Además, 3747 valores TOTP fueron generados una sola vez, es decir, el 37,47% de 10.000 valores en total. Estos resultados se basan en realizar pruebas únicamente con valores de cuatro dígitos de longitud, debido a su facilidad de trabajo en términos de análisis y visualización.

Estos resultados nos muestran que tanto la generación de HOTP y TOTP manejan algoritmos de aleatoriedad que garantizan que cada código generado sea único y casi imposible de adivinarse sin acceso a la llave compartida.

La realización de pruebas de usabilidad en el sistema de autenticación con OTP basado en el tiempo (TOTP) fue de vital importancia, ya que este tipo de sistema se utiliza en muchos entornos de seguridad y es fundamental que los usuarios puedan utilizarlo de manera eficiente y sin problemas. Una prueba de usabilidad adecuada permitirá evaluar la facilidad de uso y la eficacia del sistema, identificar posibles problemas y mejorar la experiencia de usuario. Además, las pruebas de usabilidad también pueden ayudar a identificar los errores y las dificultades que los usuarios pueden encontrar al utilizar el sistema, lo que permite a los desarrolladores mejorar y optimizar el sistema en función de las necesidades de los usuarios.

El 44% de los encuestados detectaron problemas apenas estéticos y de menor usabilidad, mientras que el 12% tuvo problemas de mayor usabilidad respecto a la heurística H1. El 63% de los encuestados detectaron problemas apenas estéticos y el 37% de menor usabilidad respecto a la heurística H2. El 56% de los encuestados detectaron problemas apenas estéticos y el 44% de menor usabilidad respecto a la heurística H3. El 69% de los encuestados detectaron problemas apenas estéticos y el 31% de menor usabilidad respecto a la heurística H4. En la heurística H5 el 31% de los encuestados no encontraron problemas de usabilidad, el 50% encontraron problemas estéticos y el 19% problemas menores. El 56% encontró problemas menores a comparación del 6% que no encontró problemas de usabilidad respecto a la heurística H6. En la heurística H7 el 19% encontró problemas mayores de usabilidad, lo cual se refiere a aceleradores para una rápida interacción con el usuario. El 56% de los encuestados no encontró problemas respecto a la heurística H8, que indica que la interfaz es sencilla y fácil de entender. El 69% encontró problemas apenas estéticos respecto a la heurística H9. Por último, el 31% de los encuestado encontró problemas mayores de usabilidad respecto a las heurística H10, lo cual se refiere a que no existe una ayuda adecuada mediate videos y guías al usuario. Los resultados por cada heurística se detallan en el Anexo VII.

## 5.2 Conclusiones

En el proceso de revisión de literatura, se pudo observar distintos algoritmos de generación de OTP, de los cuales, HOTP y en particular TOTP son los algoritmos de generación de OTP más utilizados para la autenticación de doble factor. Si bien presentan varias similitudes, existen diferencias importantes que son tomadas en cuenta a la hora de escoger uno de estos algoritmos para su implementación en un sistema. TOTP está basado en HOTP en dónde en vez de usar un contador se utiliza una marca de tiempo para la generación del OTP. Esto significa que en HOTP el código es válido hasta que sea usado, mientras que en TOTP los códigos caducan dado un intervalo de tiempo sin importar si han sido usados o no.

En el proceso de generación de TOTP, se pudo notar un aspecto importante con relación al índice de repetición de las contraseñas. El número total de combinaciones posibles de OTP con seis dígitos, es igual a un millón. Tomando en cuenta que el intervalo de tiempo para la generación de OTPs, por lo general, es de 30 segundos, y suponiendo que ningún valor de OTP se repita, todas las posibles combinaciones de seis dígitos se acabarían mostrando en un tiempo de 347,2 días. Por lo tanto, aunque se sabe que los códigos se van a repetir, utilizar HOTP o TOTP sigue siendo una forma segura de autenticación.

En el proceso de implementación de TOTP en el prototipo del sistema de autenticación, se hizo evidente la importancia de trabajar con herramientas de fácil uso y una comunicación apropiada y eficiente. Dado que los intervalos de tiempo para la generación, envío y validación de OTP son relativamente cortos, se requiere que no existan demoras en las comunicaciones ni en los procesos. El uso de herramientas como Python y Flask fue debido a su alta compatibilidad y facilidad de uso.

Por último, independientemente del tipo de OTP utilizado, su implementación en un sistema de autenticación representa una capa adicional de seguridad para el usuario. El uso correcto de los OTP reduce positivamente las posibilidades de ser víctimas de delitos informáticos como el fraude, robo de información y estafa.

## 5.3 Recomendaciones

Durante el transcurso del proyecto se trabajó con la librería open source PyOTP de Python para la generación de OTPs. Si se requiere trabajar con otras librerías o generadores de OTP se recomienda elegir las que cumplan con los requisitos mínimos de seguridad y

priorizar las que sean de código abierto para su implementación personalizada en sistemas de autenticación.

Dado que el algoritmo de funciones hash SHA-1 ha sufrido ya ataques en el pasado, se recomienda hacer uso de funciones hash con mayores tamaños de bloque como puede ser SHA-256 o SHA-512. Para unas mejores prácticas de autenticación se recomienda seguir la OWASP Authentication Cheat Sheet y la Guía de Autenticación Digital de la NIST.

Dado los resultados de las evaluaciones de usabilidad, se recomienda realizarlas a un público más amplio con una versión mejorada del prototipo tomando en cuenta las falencias encontradas. A pesar de que este tipo de pruebas son comunes para el diseño de sitios web, también pueden ser aplicadas a otros productos software de escritorio y dispositivos móviles.

Por último, el período de tiempo recomendado durante el cual el OTP es válido es de 30 segundos si la cantidad de dígitos no excede a seis. Un período de tiempo menor a 30 segundos puede causar dificultad al usuario al momento de ingresar el OTP para su respectiva validación.

# 6 REFERENCIAS BIBLIOGRÁFICAS


[1] K. Aravindhan, “One-time Password: A Survey,” *International Journal of Emerging Trends in Engineering and Development Issue 3*, vol. 1, no. 3, 2013.

[2] Mohamed. H. S. AbouSteit, A. F. Tammam, and A. Wahdan, “A Novel Approach For Generating One-Time Password With Secure Distribution,” in *2020 Fourth World Conference on Smart Trends in Systems, Security and Sustainability (WorldS4)*, Jul. 2020, pp. 461–466. doi: 10.1109/WorldS450073.2020.9210322.

[3] L. Lamport, “Password authentication with insecure communication,” *Commun ACM*, vol. 24, no. 11, pp. 770–772, Nov. 1981, doi: 10.1145/358790.358797.

[4] S. Srivastava and M. Sivasankar, “On the generation of alphanumeric one time passwords,” in *2016 International Conference on Inventive Computation Technologies (ICICT)*, Aug. 2016, pp. 1–3. doi: 10.1109/INVENTIVE.2016.7823287.

[5] M. H. Eldefrawy, K. Alghathbar, and M. K. Khan, “OTP-Based Two-Factor Authentication Using Mobile Phones,” in *2011 Eighth International Conference on Information Technology: New Generations*, Apr. 2011, pp. 327–331. doi: 10.1109/ITNG.2011.64.

[6] S. P. Shyry, M. Mahithaasree, and M. Saranya, “Implementation of One Time Password by 3 3 Vedic Multiplier,” in *2018 International Conference on Computer, Communication, and Signal Processing (ICCCSP)*, Feb. 2018, pp. 1–5. doi: 10.1109/ICCCSP.2018.8452861.

[7] C.-S. Park, “One-time password based on hash chain without shared secret and re-registration,” *Comput Secur*, vol. 75, pp. 138–146, Jun. 2018, doi: 10.1016/j.cose.2018.02.010.

[8] R. Liu, X. Wang, and C. Wang, “An efficient two-factor authentication scheme based on negative databases: Experiments and extensions,” *Appl Soft Comput*, vol. 119, p. 108558, Apr. 2022, doi: 10.1016/j.asoc.2022.108558.

[9] H. S. Elganzoury, A. A. Abdelhafez, and A. A. Hegazy, “A new secure one-time password algorithm for mobile applications,” in *2018 35th National Radio Science Conference (NRSC)*, Mar. 2018, no. Nrsc, pp. 249–257. doi: 10.1109/NRSC.2018.8354370.

[10] D. M’Raihi, M. Bellare, F. Hoornaert, D. Naccache, and O. Ranen, “HOTP: An HMAC-Based One-Time Password Algorithm,” Dec. 2005. doi: 10.17487/rfc4226.

[11] H. Krawczyk, M. Bellare, and R. Canetti, “HMAC: Keyed-Hashing for Message Authentication,” Feb. 1997. doi: 10.17487/rfc2104.

[12] Lina Lumburovska, Jovana Dobreva, Stefan Andonov, Hristina Mihajloska Trpcheska, and Vesna Dimitrova, “A Comparative Analysis of HOTP and TOTP Authentication Algorithms. Which one to choose?,” *International Scientific Journals of Scientific Technical Union of Mechanical Engineering “Industry 4.0,”* vol. 5, no. 4, pp. 131–136, 2021.

[13] D. M’Raihi, S. Machani, M. Pei, and J. Rydell, “TOTP: Time-Based One-Time Password Algorithm,” May 2011. doi: 10.17487/rfc6238.

[14] C. Sudar, S. K. Arjun, and L. R. Deepthi, “Time-based one-time password for Wi-Fi authentication and security,” in *2017 International Conference on Advances in Computing,*

*Communications and Informatics (ICACCI)*, Sep. 2017, pp. 1212–1216. doi: 10.1109/ICACCI.2017.8126007.

[15] E. Huseynov and J.-M. Seigneur, "Hardware TOTP tokens with time synchronization," in *2019 IEEE 13th International Conference on Application of Information and Communication Technologies (AICT)*, Oct. 2019, pp. 1–4. doi: 10.1109/AICT47866.2019.8981762.

[16] M. Gerami and S. Ghiasvand, "One-Time Passwords via SMS," *Bulletin de la Société Royale des Sciences de Liège*, pp. 106–113, Jun. 2016, doi: 10.25518/0037-9565.5208.

[17] X. Ye, W. Wen, Y. Ye, and Q. Cen, "An OTP-Based Mechanism for Defending Application Layer DDoS Attacks," in *CCIS*, vol. 227, 2011, pp. 388–396. doi: 10.1007/978-3-642-23226-8_51.

[18] S. A. Lone and A. H. Mir, "A novel OTP based tripartite authentication scheme," *International Journal of Pervasive Computing and Communications*, vol. 18, no. 4, pp. 437–459, 2022, doi: 10.1108/IJPCC-04-2021-0097.

[19] R. A. Grimes, "One-Time Password Attacks," in *Hacking Multifactor Authentication*, Wiley, 2020, pp. 205–226. doi: 10.1002/9781119672357.ch9.

[20] R. A. Grimes, "Types of Authentication," in *Hacking Multifactor Authentication*, 2021, pp. 59–99. doi: 10.1002/9781119672357.ch3.

[21] S. M. Mitchell and C. B. Seaman, "A comparison of software cost, duration, and quality for waterfall vs. iterative and incremental development: A systematic review," in *2009 3rd International Symposium on Empirical Software Engineering and Measurement*, Oct. 2009, pp. 511–515. doi: 10.1109/ESEM.2009.5314228.

[22] E. Linares-Espinós *et al.*, "Metodolog\'\ia de una revisión sistemática," *Actas Urol Esp*, vol. 42, no. 8, pp. 499–506, 2018.

[23] N. Haller, C. Metz, P. Nesser, and M. Straw, "A One-Time Password System," Feb. 1998. doi: 10.17487/rfc2289.

[24] K. R. Srinath, "Python–the fastest growing programming language," *International Research Journal of Engineering and Technology*, vol. 4, no. 12, pp. 354–357, 2017.

[25] M. R. Mufid, A. Basofi, M. U. H. al Rasyid, I. F. Rochimansyah, and A. rokhim, "Design an MVC Model using Python for Flask Framework Development," in *2019 International Electronics Symposium (IES)*, Sep. 2019, pp. 214–219. doi: 10.1109/ELECSYM.2019.8901656.

[26] M. Singh, A. Verma, A. Parasher, N. Chauhan, and G. Budhiraja, "Implementation of database using python flask framework," *International Journal of Engineering and Computer Science*, vol. 8, no. 12, pp. 24890–24893, 2019.

[27] A. Viloria, G. C. Acuña, D. J. Alcázar Franco, H. Hernández-Palma, J. P. Fuentes, and E. P. Rambal, "Integration of Data Mining Techniques to PostgreSQL Database Manager System," *Procedia Comput Sci*, vol. 155, pp. 575–580, 2019, doi: 10.1016/j.procs.2019.08.080.

# 7 ANEXOS

## ANEXO I. Ventajas y desventajas de protocolos OTP.

| Protocolo | Ventajas | Desventajas |
|---|---|---|
| Protocolo de funciones simples de hash | Fácil de implementar | Número finito de sesiones; los OTP son almacenados en memoria; el OTP para la siguiente sesión se envía por la red; no hay mecanismos de sincronización; autenticación unidireccional; la validez del OTP no está limitada por el tiempo |
| Protocolo de cadena hash de Lamport | Resistencia al espionaje; Presencia de mecanismos de sincronización. | Número finito de sesiones; complejidad computacional; autenticación unidireccional; la validez de los OTP no está limitada por el tiempo. |
| Protocolo YSH | Resistencia al espionaje; presencia de mecanismos de sincronización; autenticación bidireccional. | Número finito de sesiones; complejidad computacional; la validez de los OTP no está limitada por el tiempo |
| Protocolo Bicakci | Facilidad de implementación, uso de algoritmos criptográficos, resistencia a la escucha, presencia de mecanismos de sincronización, número infinito de sesiones, las OTP no se almacenan en memoria. | Autenticación unidireccional, complejidad computacional, la validez de las OTP no está limitada en el tiempo. |
| Esquema de Chefranov | Autenticación bidireccional; número infinito de sesiones | Complejidad de implementación, no hay mecanismos de sincronización, el servidor almacena una OTP para la siguiente sesión, la validez de las OTP no está limitada en el tiempo. |
| Protocolo HOTP | Presencia de mecanismos de sincronización, estadisticamente adivinar una OTP es el mejor ataque, número infinito de sesiones, las OTP no se almacenan en memoria, resistencia a la escucha. | Autenticación unidireccional, la validez de las OTP no está limitada por el tiempo. |
| Protocolo TOTP | Presencia de mecanismos de sincronización, estadisticamente adivinar una OTP es el mejor ataque, número infinito de sesiones, las OTP no se almacenan en memoria, resistencia a la escucha, la validez de las OTP está limitada en el tiempo. | Autenticación unidireccional |

# ANEXO II. Historias de Usuario.

| Historias de Usuario | | | | |
|---|---|---|---|---|
| Implementar un generador OTP basado en tiempo (TOTP) para autenticación de dos factores | | | | |
| **Código** | **Título** | **Descripción** | **Prioridad** | **Estimación** |
| | EPIC1 | Configuración inicial del generador OTP | | |
| US1.1 | Configuración de generador OTP | Como usuario, quiero poder configurar mi generador OTP para generar contraseñas basadas en tiempo | Alta | 2 |
| US1.2 | Generar clave secreta | Como usuario, quiero poder generar una clave secreta única y compartida para mi generador OTP. | Alta | 3 |
| US1.3 | Duración del tiempo de OTP | Como usuario, quiero poder elegir la duración del tiempo de cada contraseña generada. | Media | 1 |
| | EPIC2 | Generación de contraseñas OTP | | |
| US2.1 | Generar OTP | Como usuario, quiero poder generar contraseñas OTP basadas en tiempo utilizando mi generador OTP. | Alta | 2 |
| US2.2 | Generar clave secreta | Como desarrollador, quiero poder generar la clave secreta para generar contraseñas OTP. | Media | 3 |
| US2.3 | Visualización de OTP | Como usuario, quiero poder visualizar mis contraseñas OTP por un aplicativo web. | Baja | 1 |
| | EPIC3 | Integración en la aplicación de autenticación de dos factores | | |
| US3.1 | Uso del generador OTP | Como usuario, quiero poder usar mi generador OTP para autenticarme en una aplicación de autenticación de dos factores. | Alta | 3 |
| US3.2 | Configuración de app para OTP | Como administrador, quiero poder configurar la aplicación de autenticación de dos factores para utilizar un generador OTP basado en tiempo. | Media | 2 |
| US3.3 | Interagración de OTP | Como desarrollador, quiero poder integrar la generación OTP basada en tiempo en la aplicación de autenticación de dos factores. | Alta | 5 |
| | EPIC4 | Validación de contraseñas OTP | | |
| US4.1 | Validación de OTP por el usuario | Como usuario, quiero poder validar las contraseñas OTP generadas por mi generador OTP. | Alta | 2 |
| US4.2 | Validación de OTP por el administrador | Como administrador, quiero poder validar las contraseñas OTP generadas por los usuarios en la aplicación de autenticación de dos factores. | Media | 3 |
| US4.3 | Agregación de validación de OTP | Como desarrollador, quiero poder agregar la validación de contraseñas OTP en la aplicación de autenticación de dos factores. | Alta | 5 |

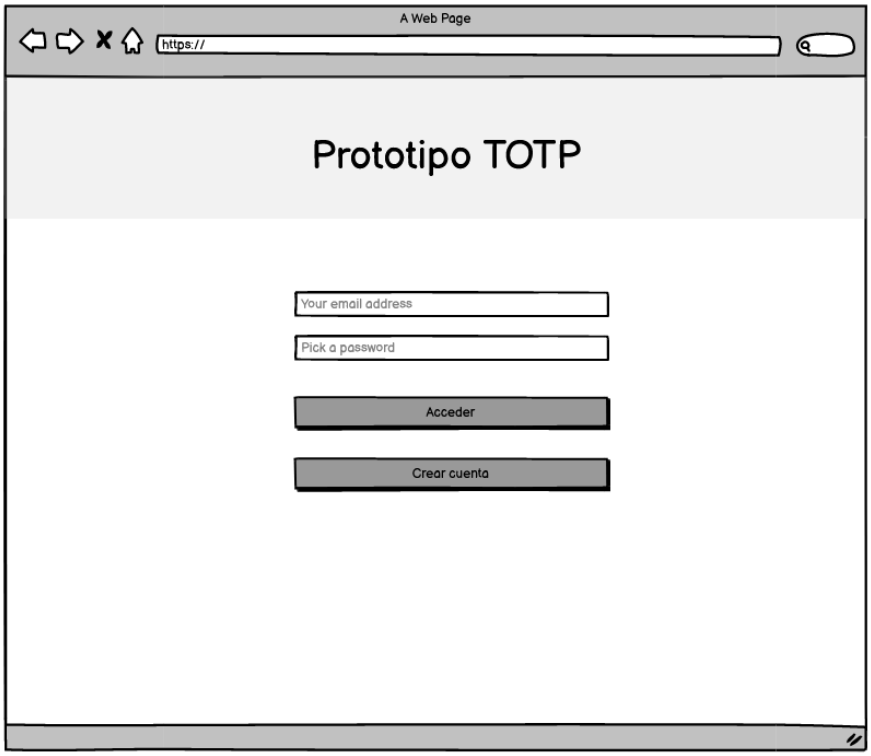
# ANEXO III. Análisis comparativo de ataques a OTP.

| Ataque | Protocolo de funciones simples de hash | Protocolo de cadena hash de Lamport | Protocolo YSH | Protocolo Bicakci | Esquema de Chefranov | Protocolo HOTP | Protocolo TOTP |
|---|---|---|---|---|---|---|---|
| Navegación por la tabla de verificación | - | + | + | + | + | + | + |
| Modificación en la tabla de verificacion | - | - | - | + | - | + | + |
| Spoofing de contraseñas que son enviadas para la siguiente sesión de autenticación | - | + | + | + | + | + | + |
| Robo de llave secreta | * | * | - | - | - | + | + |
| Interceptación de contraseñas que son enviadas para la siguiente sesión de autenticación | - | + | + | + | + | + | + |
| Ataque small number | + | - | - | + | + | + | + |
| Ataque de suplantación | - | - | - | - | - | - | - |

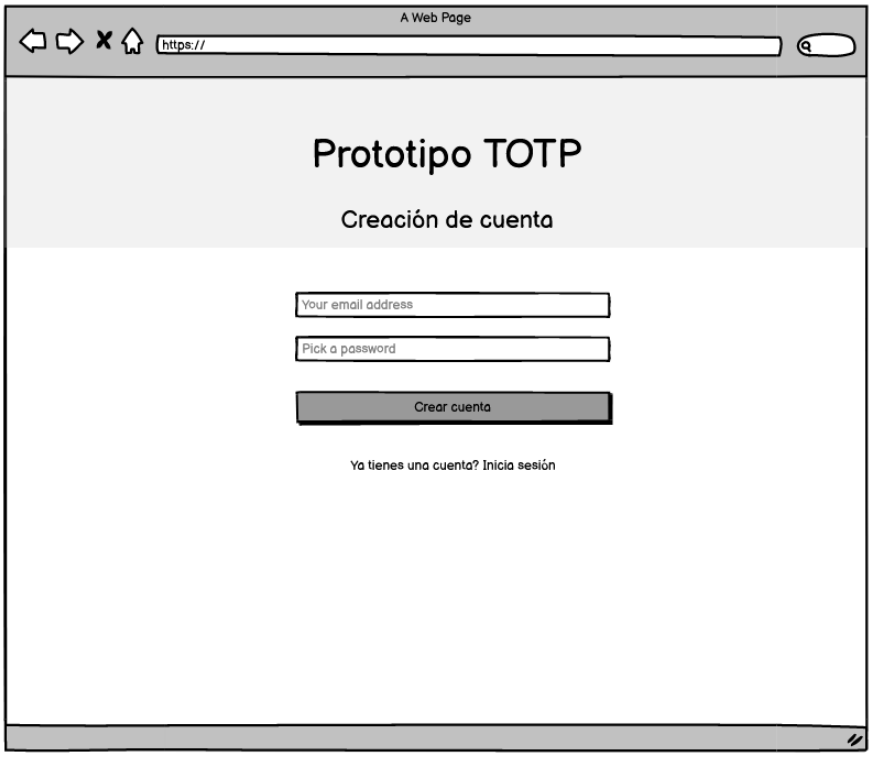
"+" significa que es resistente, "-" significa que no es resistente, "*" significa que no haya datos disponibles

# ANEXO IV. Mockups.

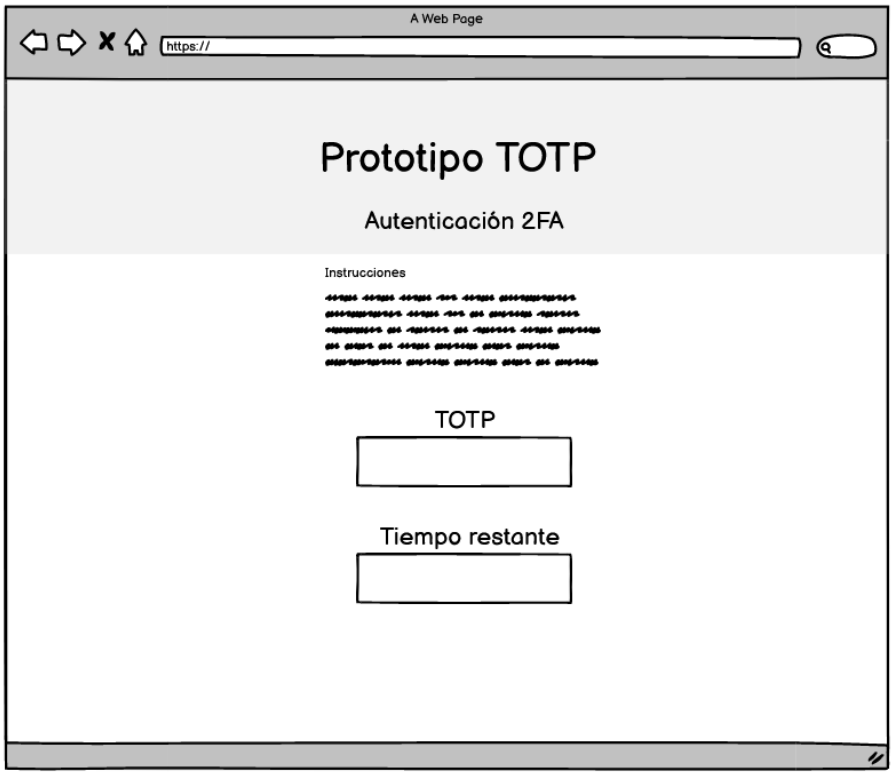
## Inicio de sesión




## Creación de cuenta



## Autenticación 2FA mediante TOTP



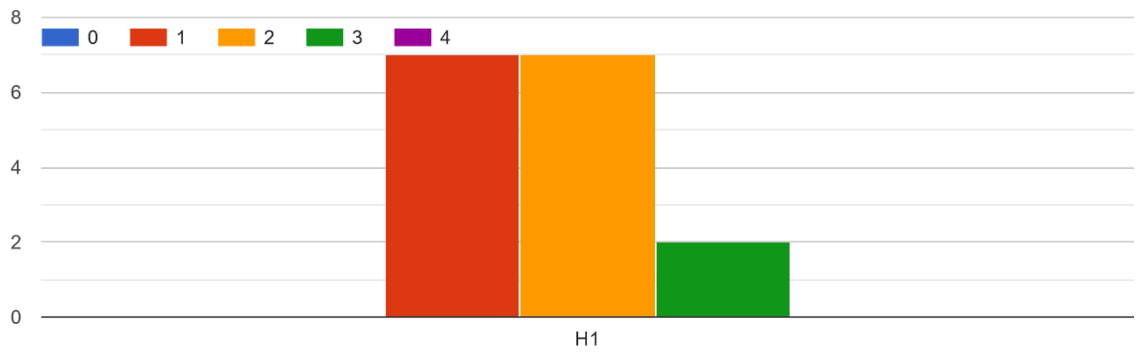

## ANEXO V. Ingreso de Autenticación

## ANEXO VI. Visualización de OTP para autenticación.


## ANEXO VII. Resultados de las Pruebas de Usabilidad.

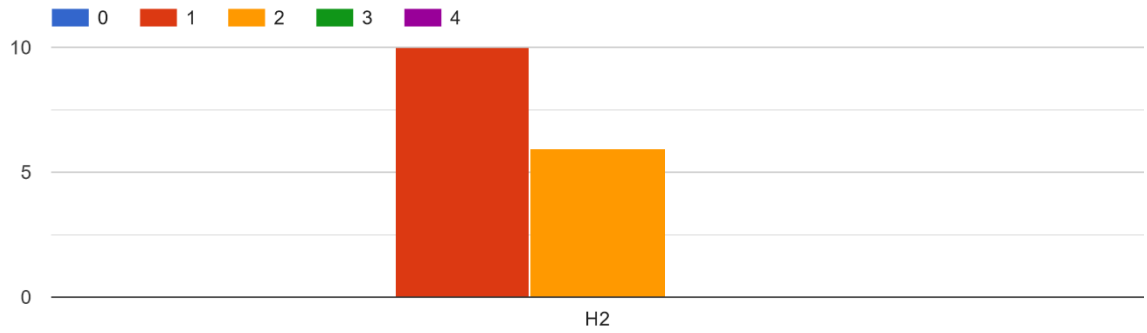
### Heurística H1

¿El sistema mantiene informados a los usuarios sobre su progreso?



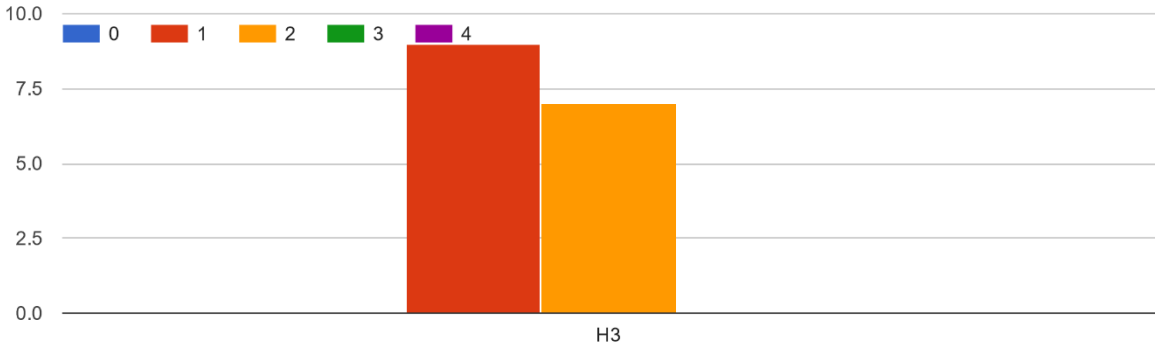
## Heurística H2

¿El sistema habla el idioma de los usuarios, con palabras, frases y conceptos familiares para el usuario, en lugar de términos orientados al sistema?



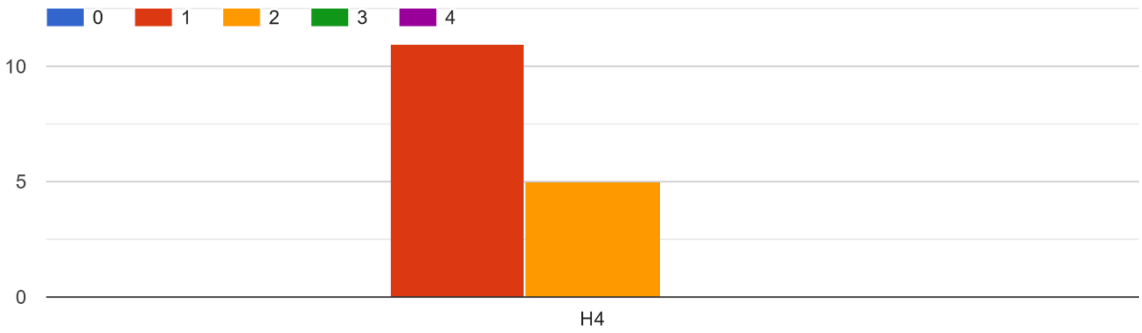

## Heurística H3

¿El sistema muestra acciones que permiten al usuario repetir, salir o retroceder una acción previamente realizada?



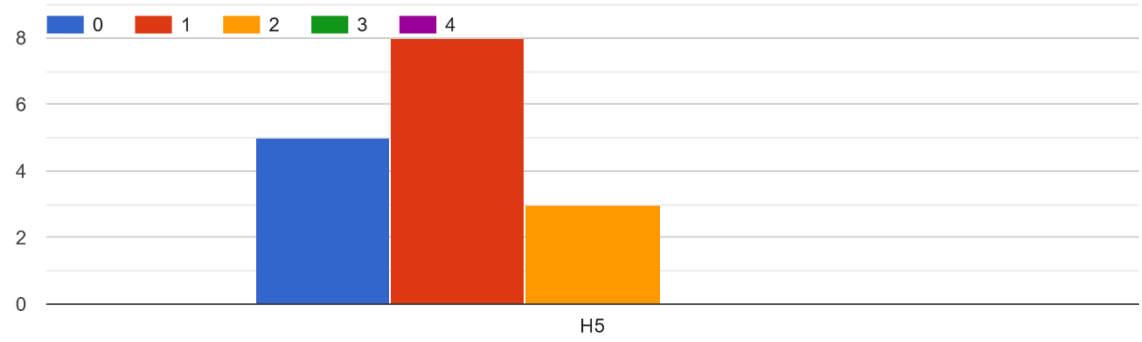
## Heurística H4

¿El sistema se maneja por un conjunto de estándares?



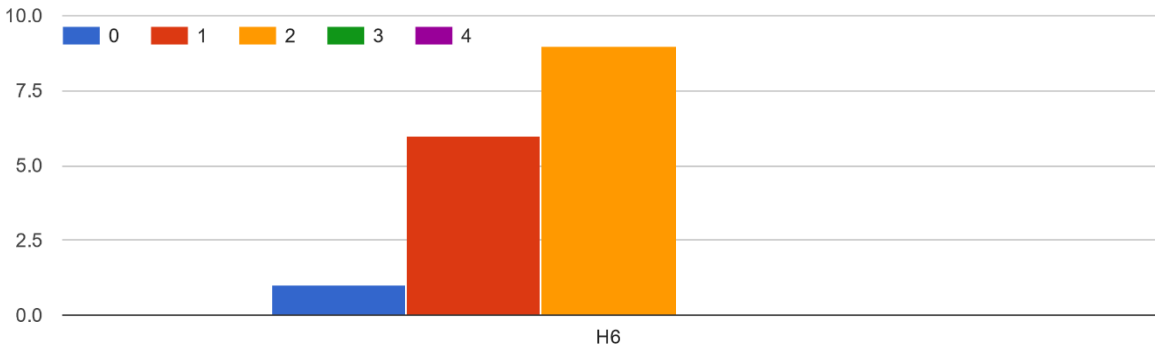

## Heurística H5

¿El sistema presenta un diseño simple y fácil de usar que ayuda a que los usuarios tengan menos incidencia a cometer errores?



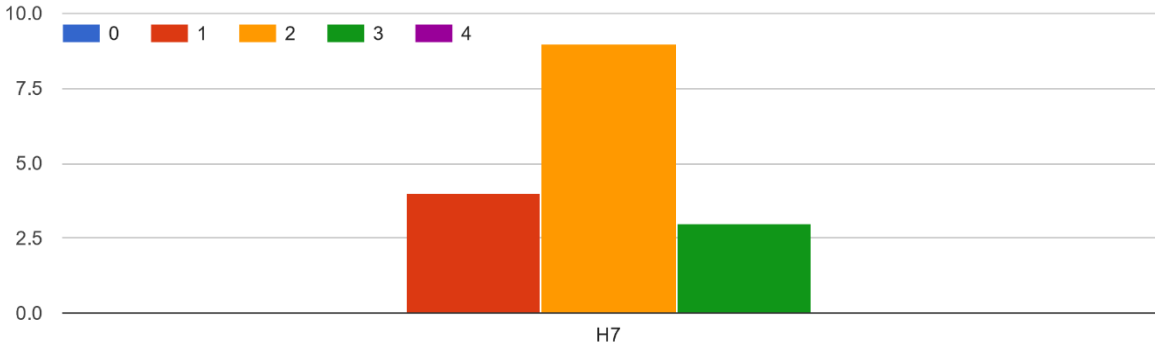
## Heurística H6

¿El sistema presenta opciones, objetos y acciones que siempre están visibles para los usuarios, para que estos no tengan que recordarlas entre las distintas secciones del sistema?



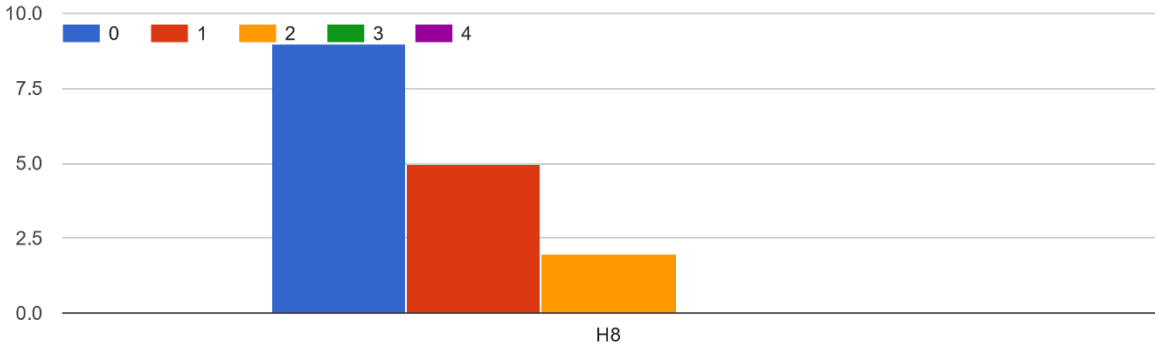

## Heurística H7

¿El sistema presenta aceleradores que permiten facilitar la interacción con el usuario?



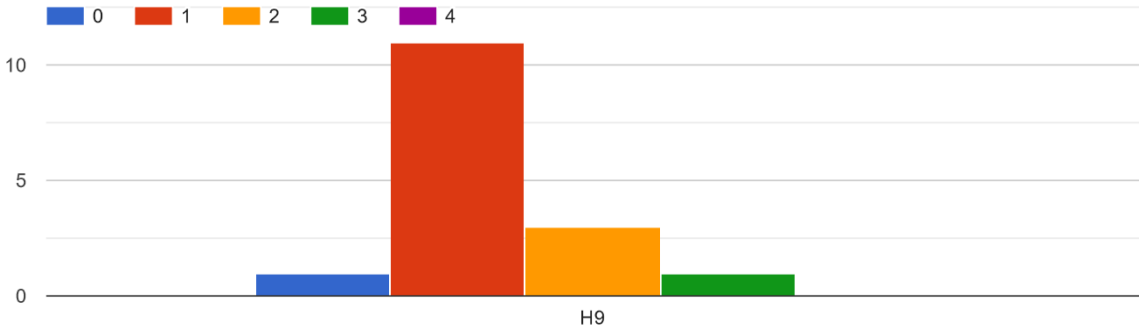
## Heurística H8

¿La interfaz del sistema es sencilla y fácil de entender?



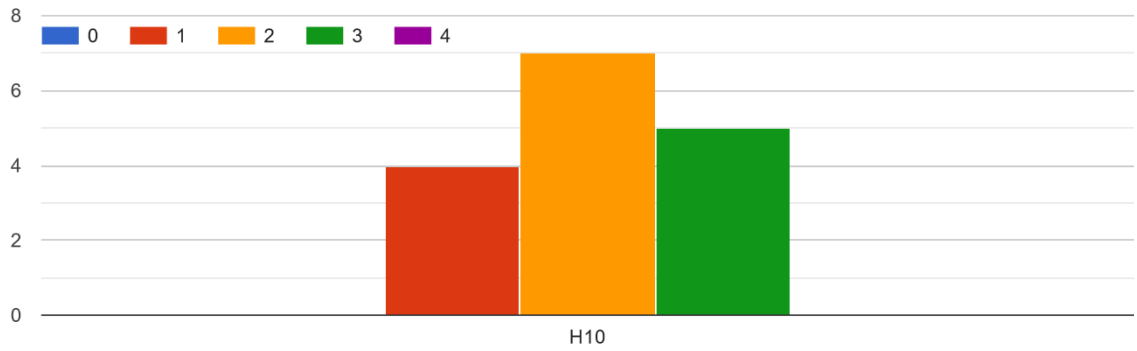

## Heurística H9

¿ Los mensajes de error se expresan en un lenguaje sencillo (sin códigos de error), indican con precisión el problema y sugieren una solución de manera constructiva ?



## Heurística H10

¿El sistema brinda ayuda adecuada y precisa, como: instrucciones, videos de ayuda, guías, ¿entre otros ?




## ANEXO VIII. Código de implementación.

Anexo. Enlace al repositorio de código