COMILLAS

UNIVERSIDAD PONTIFICIA

ICAI

GRADO EN INGENIERÍA EN TECNOLOGÍAS DE TELECOMUNICACIÓN

TRABAJO FIN DE GRADO

AUDIOMARK: Autenticación de dos factores por medio de audio

Autora: Andrea Fariña Fernández-Portillo

Director: Atilano Fernández-Pacheco Sánchez-Migallón

Madrid

Declaro, bajo mi responsabilidad, que el Proyecto presentado con el título

AudioMark: Autenticación de dos factores por medio de audio

en la ETS de Ingeniería - ICAI de la Universidad Pontificia Comillas en el curso académico 2019/20 es de mi autoría, original e inédito y no ha sido presentado con anterioridad a otros efectos.

El Proyecto no es plagio de otro, ni total ni parcialmente y la información que ha sido tomada de otros documentos está debidamente referenciada.

Fdo.: Andrea Fariña Fernández-Portillo Fecha: 10/06/2020

Autorizada la entrega del proyecto

EL DIRECTOR DEL PROYECTO

Fdo.: Atilano Fernández-Pacheco Sánchez-Migallón Fecha: 10/06/2020

COMILLAS

UNIVERSIDAD PONTIFICIA

ICAI

GRADO EN INGENIERÍA EN TECNOLOGÍAS DE TELECOMUNICACIÓN

TRABAJO FIN DE GRADO

AUDIOMARK: Autenticación de dos factores por medio de audio

Autora: Andrea Fariña Fernández-Portillo

Director: Atilano Fernández-Pacheco Sánchez-Migallón

Madrid

Agradecimientos

Quiero agradecer a mi director Atilano, quien me ha guiado en la consecución de este proyecto con un gran esfuerzo.

Gracias a todos mis compañeros y amigos que han recorrido este camino junto a mi durante estos cuatro años.

Además, quiero agradecer especialmente a mis padres, sin quienes no podría haber ni siquiera imaginado estar donde estoy.

AUDIOMARK: Autenticación de dos factores por medio de audio

Autora: Fariña Fernández-Portillo, Andrea.
Director: Fernández-Pacheco Sánchez-Migallón, Atilano.
Entidad Colaboradora: Cambridge Consultants

RESUMEN DEL PROYECTO

En este proyecto se ha desarrollado un entorno en el que poner en práctica el uso de un método de segundo factor de autenticación que facilita su implementación porque simplifica el proceso haciendo uso de audio.

Palabras clave: TOTP, A2F, SDK.

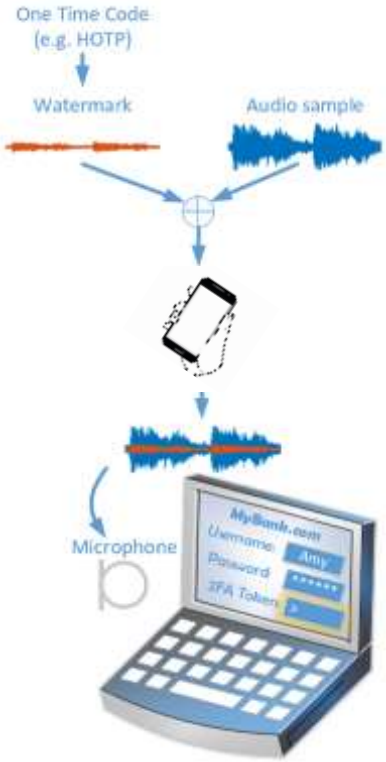
1. Introducción

Los sistemas de autenticación son necesarios para proteger información sensible de los usuarios. Por ello, la seguridad es un requisito imprescindible en estos métodos. Para mejorar la seguridad surgen la autenticación de dos factores, con la que se añade una capa adicional de seguridad. Sin embargo, muchos usuarios deciden no implementarla por ser un proceso bastante tedioso. Pero esto es un problema hoy en día cuando más y más aumentan los robos de contraseñas, que forman la primera, y en muchos casos única, capa de seguridad de las cuentas de los usuarios.

2. Definición del proyecto

Con AudioMark se pone en práctica un método de autenticación de dos factores más cómodo y entretenido para el usuario, pues el código de seis dígitos, que normalmente en los procesos de autenticación de dos factores se recibe por SMS, se encripta en un audio. De esta manera se elimina la necesidad de memorizar y posteriormente escribir el código.

Para llevar a cabo este método se realizará el proceso mostrado en el siguiente diagrama hardware:



3. Descripción del modelo/sistema/herramienta

El desarrollo de esta plataforma consta de varios componentes, relacionados entre sí. Estos se muestran en el siguiente diagrama software:

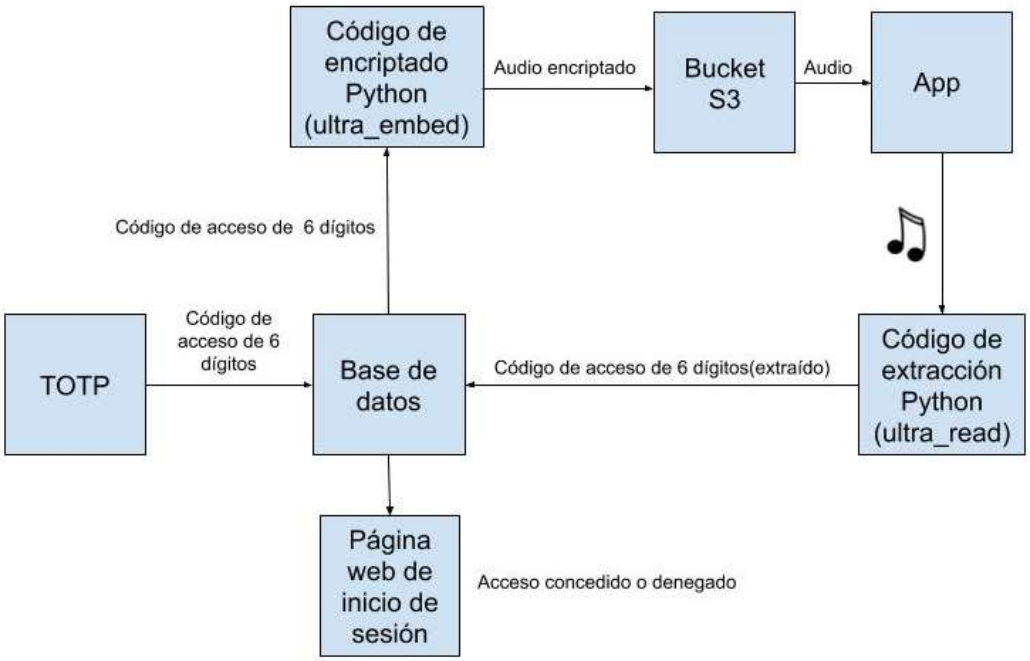


Ilustración 1 – Diagrama software del sistema

Todo esto genera un entorno que consta de un verificador web y una aplicación móvil para simular la autenticación.

4. Resultados

Se comprueba el correcto funcionamiento de AudioMark en una amplia variedad de entornos ruidosos cumpliendo los objetivos del proyecto, que eran:

- Encontrar un método para insertar un código de verificación en un audio. Este deberá ser lo suficientemente robusto como para soportar el ruido ambiental o la reproducción del audio a cierta distancia del verificador.

- Suministrar la seguridad propia de un método de autenticación, es decir, proteger el código que actúa como segundo factor de autenticación para que no pueda ser replicado.

- Construcción del entorno tecnológico para llevar a cabo la verificación por audio en dos pasos.

5. Conclusiones

AudioMark supone el comienzo de una alternativa de segundo factor de autenticación, que con las debidas modificaciones se podría utilizar en el mundo real, tanto en empresas como para el uso particular de usuarios. Facilita la implantación de la capa adicional de seguridad en las cuentas de los usuarios, dando así más confidencialidad a los datos de estos.

6. Referencias

[1] Comunicados de prensa, Kaspersky. https://www.kaspersky.es/about/press-releases/2019_estan-tus-contrasenas-a-salvo-aumenta-en-un-60-los-usuarios-afectados-por-el-robo-de-contrasenas-en-2019

AUDIOMARK: Autenticación de dos factores por medio de audio

Author: Fariña Fernández-Portillo, Andrea.
Supervisor: Fernández-Pacheco Sánchez-Migallón, Atilano.
Collaborating Entity: Cambridge Consultants

ABSTRACT

In this project an environment has been developed in which to put into practice the use of a second factor authentication method that facilitates its implementation because it simplifies the process by making use of audio.

Keywords: TOTP, 2FA, SDK

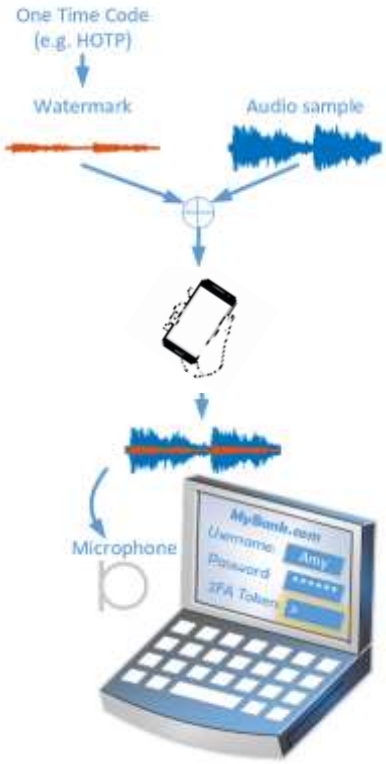
1. Introduction

Authentication systems are necessary to protect sensitive user information. Therefore, security is an essential requirement in these methods. In order to improve security, two-factor authentication emerges, adding a layer of security. However, many users decide not to implement it because it is a rather tedious process. But this is a problem today as more and more passwords are being stolen, because passwords are the first, and in many cases only, security layer of user accounts.

2. Project definition

With AudioMark, a more comfortable and entertaining two-factor authentication is implemented for the user since the six-digit code, which is normally received by SMS in two-factor authentication processes, is encrypted in audio. This eliminates the need to memorize and then write the code.

To carry out this method, the process shown in the following hardware diagram will be carried out:



3. Model/system/tool description

The development of this platform consists of several components, related to one another. These are shown in the following software diagram:

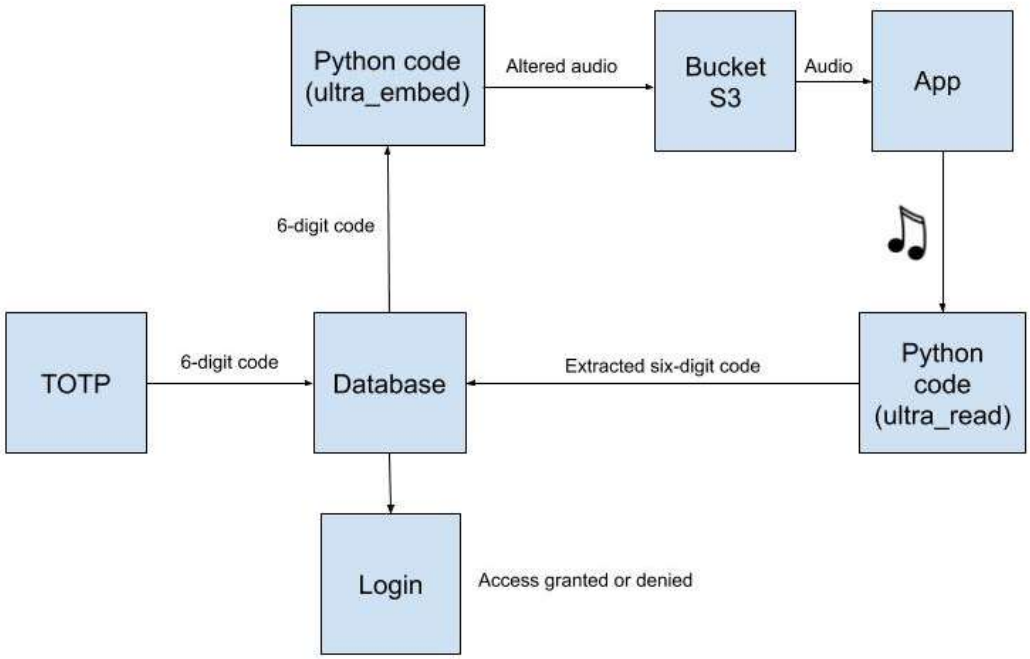


Illustration 2 – System software diagram

All this generates an environment that consists of a web verifier and a mobile application to simulate the authentication process.

4. Results

AudioMark is tested for correct operation in a wide variety of noisy environments, meeting the objectives of the project, which were:

- Find a method to insert a verification code in an audio. This method should be robust enough to withstand ambient noise or audio playback at a distance from the tester.

- Provide the security of an authentication method, that is, protect the code that acts as the second factor of authentication so that it cannot be replicated.

- Construction of the technological environment to carry out the audio authentication in two steps.

5. Conclusions

AudioMark is the beginning of a second factor authentication alternative, which with the appropriate modifications could be used in the real world, both in companies and for the private use of users. Facilitates the implementation of the additional layer of security in user accounts, thus giving more confidentiality to their data.

6. References

[1] Comunicados de prensa, Kaspersky. https://www.kaspersky.es/about/press-releases/2019_estan-tus-contrasenas-a-salvo-aumenta-en-un-60-los-usuarios-afectados-por-el-robo-de-contrasenas-en-2019

Índice de la memoria

Índice de figuras

Índice de tablas

1. INTRODUCCIÓN

La definición oficial de marca de agua según el Diccionario de la Real Academia Española es "código de identificación, generalmente imperceptible, que contiene información sobre el origen, la autoría y las condiciones de utilización de algunos soportes informáticos, para evitar usos fraudulentos". Normalmente, estas marcas son más conocidas en su uso en imágenes. Sin embargo, una marca de agua digital es una técnica que se puede aplicar sobre cualquier objeto digital (imágenes, vídeo, texto, software, audio…). La marca de agua es una técnica esteganográfica de ocultación de información.

Figura 1: Imagen con la etiqueta "2006", de manera degradada y visible.

El término "esteganografía" proviene del griego *steganos* que significa oculto o cubierto y de *graphos* que significa escritura. Consiste en la aplicación de técnicas que permiten ocultar mensajes u objetos dentro de otros, llamados portadores, para que puedan ser enviados sin que se perciba. Procura un canal encubierto de comunicación en el que los observadores que tienen acceso a este, no perciban esa comunicación.

Aunque hasta el siglo XVI no se acuñó el vocablo "estenografía", este conjunto de técnicas ha estado presente desde tiempos antiguos y ha ido evolucionando a lo largo de los años. La primera alusión al uso de la esteganografía tiene lugar en *Las historias* de Heródoto (año 403 a. C.) en el que un amo rapó la cabeza a uno de sus esclavos y le tatuó un mensaje en el cuero cabelludo, después esperó a que le volviera a crecer el cabello y lo envió al receptor del mensaje, con instrucciones de que le raparan la cabeza.

Aunque la esteganografía pueda ser confundida con la criptografía porque ambas son técnicas de protección de información, es importante saber que se trata de disciplinas diferentes. La criptografía se utiliza para codificar información con el objetivo de que sea ininteligible a simple vista y la esteganografía oculta la información para que no conste siquiera de la existencia de esta información.

Sin embargo, ambos términos pueden complementarse, cifrando la información y a su vez haciendo imperceptible su envío. La aplicación de estas tecnologías es muy útil, sobre todo en estos días por la gran cantidad de datos y la cada vez mayor importancia de mantenerlos seguros.

Con la aparición de Internet en 1969 y su desarrollo hasta la actualidad son cada vez más los datos compartidos en la Web. Se calcula que actualmente existen unos 40 zettabytes de datos y que este número se multiplicará por 4 en cinco años.

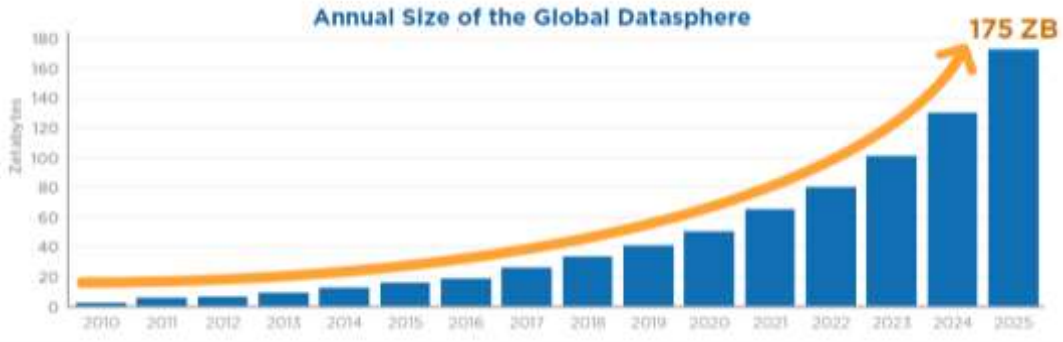


Figura 2: Tamaño global de datos anual.

Gran parte de estos son datos sensibles como información personal de usuarios, cuentas bancarias, lugar de residencia, etc. Estos datos tienen gran valor y su utilización ilícita o robo ha sido noticia en muchas ocasiones. Para proteger nuestra información surge la necesidad del uso de la autenticación.

La autenticación en la seguridad digital se define como el procedimiento de comprobación de la identidad de un usuario de un recurso informático. El usuario que quiere acceder a ciertos datos o recursos se denomina probador y el sistema que protege el acceso a dichos recursos y que se encarga de verificar que el probador es un usuario con permisos para acceder a dichos recursos es el verificador.

Existen diferentes métodos de autenticación en función de qué se utilice para la verificación de identidad. Se dividen en tres categorías diferentes:

- Sistemas basados en algo conocido como una contraseña.
- Sistemas basados en algo poseído como una tarjeta de identidad.
- Sistemas basados en una característica física de un usuario como la verificación de voz.

Sin embargo, la seguridad de los datos nunca está garantizada por completo. Según Kaspersky, compañía internacional dedicada a la ciberseguridad, los robos de contraseñas aumentaron en un 60% de 2018 a 2019 [1].

Como una mejora a la seguridad que ofrece el popular uso de identificación y contraseña, aparece el múltiple factor de autenticación. La autenticación de múltiples factores es un método de control de acceso informático en el que el usuario se le concede acceso al sistema tras presentar dos (en el caso de autenticación de dos factores) o más pruebas diferentes que le autentiquen. Existen diferentes tipos de múltiple factor de autenticación, que siempre combinan métodos de los tres tipos de autenticación diferentes.

La autenticación de dos factores (A2F) combina normalmente algo que se conoce, con algo que se tiene. Es imposible para un ladrón de contraseñas conseguir acceder a los datos de su víctima porque, aunque haya averiguado su contraseña, necesitará también el segundo factor de autenticación. Un ejemplo muy común de A2F es el uso de una contraseña seguida de un código de verificación recibido por SMS. Este es en la actualidad el método más extendido para acceder a cuentas de correo como Gmail, acceso a cuentas bancarias o realizar otras operaciones vía telemática, pero por lo general, esta capa de protección adicional no se activa automáticamente, sino que es el usuario quien debe solicitarla.

La gran desventaja de A2F es que su ergonomía de utilización resulta demasiado tediosa para la mayoría de usuarios, y por eso deciden no implementarla en sus procesos de autenticación. Además, el ritmo frenético de hoy en día hace que el proceso de tener que esperar a recibir un SMS de verificación, tener que memorizarlo o escribirlo, e

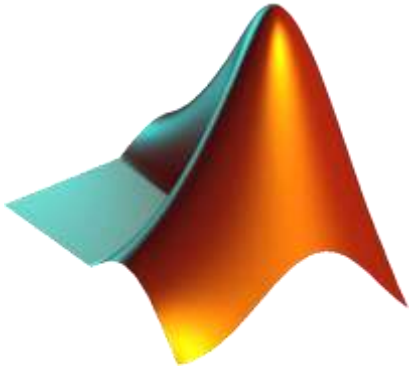
introducirlo en el verificador, sea rechazado por los usuarios, quienes prefieren la rapidez y comodidad a añadir una segunda capa de seguridad a sus datos.

Cada vez se están creando más métodos de implementación del segundo factor de autenticación, ofreciendo diferentes alternativas. En febrero de 2020 se implementó A2F en el verificador del portal de acceso personal de Boston University, ofreciendo Duo Security, una llamada telefónica o una notificación *push* como las tres alternativas para el segundo factor de autenticación. Aun así, la mayoría de alumnos consideran el proceso bastante lento y aburrido.

La combinación de la esteganografía y la criptografía aplicadas a un elemento más atractivo para el usuario como puede ser un audio, puede dar como resultado un método menos tedioso de autenticación de dos factores, que satisfaga las necesidades de seguridad sin suponer un peso que los usuarios prefieran descartar.

2. DESCRIPCIÓN DE LAS TECNOLOGÍAS

2.1 MATLAB



MATLAB [2] es un sistema de cómputo numérico desarrollado por MathWorks que ofrece un entorno de desarrollo integrado (IDE) con un lenguaje de programación propio, lenguaje M, destinado sobre todo a computación científica. Este lenguaje es interpretado, y puede ejecutarse tanto en el entorno interactivo, como a través de un archivo de script (archivos *.m).

Este entorno de programación se utiliza mucho en campos técnicos para el análisis de datos, la resolución de problemas y para la experimentación y el desarrollo de algoritmos. Destaca su uso en centros de investigación y desarrollo y en universidades.

Las prestaciones de MATLAB son muy amplias gracias a dos herramientas fundamentales: una plataforma de simulación multidominio (Simulink) y un editor de interfaces de usuario (GUIDE). Además, las cajas de herramientas de MATLAB, llamadas *toolboxes* amplían las capacidades de este software matemático.

En el desarrollo de este proyecto cabe destacar la utilización de una caja de herramientas para el procesamiento de audio, el análisis del habla y la medición acústica, llamado Audio Toolbox™. También proporciona algoritmos para la extracción de funciones de audio y voz y la transformación de la señal de audio.

MATLAB está disponible para las plataformas de Unix, Windows, macOS y GNU/Linux y es un software propietario gratuito.

2.2 Visual Studio Code

Visual Studio Code [3] es un editor de código fuente desarrollado por Microsoft para Windows, Linux y macOS. Se trata de una plataforma a través de la cual se pueden escribir programas en cualquier lenguaje de programación, permitiendo descargar diferentes tipos de componentes y extensiones, que ayudan a la edición, permiten resaltar el texto que pueda estar mal escrito, corrigen errores y aportan la finalización inteligente de código. Además, Visual Studio Code permite recorrer los diferentes documentos y carpetas de un proyecto, a través del explorador de archivos.

Visual Studio Code combina la simplicidad y el carácter intuitivo de un editor de código fuente con potentes herramientas de desarrollador, como la finalización y depuración de código IntelliSense. Esto permite que sea utilizado de forma sencilla por el usuario para desarrollar el código.

Entre las ventajas de este editor de código destaca la posibilidad de emplear Git, que permite trabajar con el código fuente controlando qué cambios se han realizado y guardarlos sin salir del editor.

Finalmente, es importante añadir que Visual Studio Code es un software gratuito, de código abierto y libre distribución, aunque su descarga oficial está bajo software propietario e incluye características personalizadas por Microsoft.

2.3 Amazon Web Service (AWS)

Amazon es una compañía conocida generalmente por sus servicios de comercio electrónico. Sin embargo, hay otra parte de Amazon que es probablemente menos conocida pero que es responsable de una décima parte de los ingresos de la compañía y que produce más ganancias netas que cualquier otro sector de Amazon. Se trata de Amazon Web Service [4], una colección de servicios de computación en la nube pública, o servicios web, que en conjunto forman una plataforma de computación en la nube, ofrecida a través de Internet por Amazon.com.

Amazon Web Services ofrece una amplia variedad de servicios desde servicios informáticos como Amazon Elastic Compute Cloud (Amazon EC2), que permite a los usuarios alquilar computadores virtuales en los cuales ejecutar sus propias aplicaciones, hasta seguridad, identidad y conformidad como AWS Identity and Access Management (IAM) que controlan el acceso a los servicios de AWS mediante la creación de políticas de acceso, usuarios y grupos.

Además, AWS ofrece servicios de almacenamiento en la nube que garantizan durabilidad, disponibilidad y escalabilidad. Este servicio se denomina Amazon Simple Storage Service o Amazon S3 y permite almacenar cualquier cantidad de datos desde cualquier lugar.

Esta plataforma permite a las empresas o particulares comprar espacio de almacenamiento para almacenar una gran base de datos, ancho de banda para alojar un sitio web o poder de procesamiento para ejecutar software complejo de forma remota. Entre ellos destaca Netflix, Airbnb o GE, que utilizan Amazon S3 para almacenar y distribuir miles de millones de datos a todo el mundo de forma segura.

Amazon S3 almacena datos a modo de objetos dentro de buckets, que son contenedores de objetos. Estos objetos se componen de un archivo y, de forma opcional, de cualquier metadato que aporte una descripción del archivo. Para almacenar un objeto en Amazon S3, se debe cargar en un bucket el archivo que se quiera almacenar. Al hacer esto, se configurarán los permisos en el objeto y en cualquier metadato.

Para que las transferencias entre archivos entrantes y salientes de Amazon S3 sean eficientes, AWS consta de una interfaz de línea de comandos (CLI) que presenta un conjunto de simples comandos de archivo. Además, esta interfaz de línea de comandos es una herramienta unificada para administrar productos y controlar y automatizar servicios de AWS.

Entre las mayores ventajas de AWS destaca su simplicidad, fácil acceso global y gestión de tareas que permiten crear una estructura back-end mediante la consola de administración de AWS, que es una sencilla e intuitiva interfaz web, sin necesidad de hardware adicional.

2.4 Node.js

Node.js [5] es un entorno en tiempo de ejecución multiplataforma que permite el desarrollo de la capa del servidor. Está basado en el lenguaje de programación JavaScript, asíncrono, con periférico de entrada/salida de datos en una arquitectura orientada a eventos.

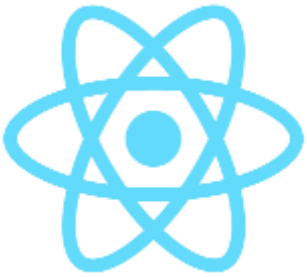
Su origen fue crear un entorno enfocado a la creación de programas de red altamente escalables, como servidores web.

Cabe destacar que es una tecnología gratuita y de código abierto compatible con Windows, macOS, Linux, Solaris, FreeBSD, OpenBSD y webOS. Tiene un alto rendimiento y gran facilidad de uso y aprendizaje.

2.5 NPM

NPM (Node Packet Manager) [6] es el administrador de paquetes predeterminado para el entorno de tiempo de ejecución de JavaScript Node.js. Coloca los módulos en su lugar para que Node pueda encontrarlos y gestiona los conflictos de dependencia de forma inteligente. En este proyecto ha sido empleado junto a Node para el despliegue de los archivos necesarios para el desarrollo de la aplicación móvil.

2.6 React Native



React Native [7] es un marco de aplicaciones móvil en JavaScript de código abierto creado por Facebook. Se utiliza para desarrollar aplicaciones para iOS y Android, aunque también es compatible con UWP. Permite a los desarrolladores usar la biblioteca JavaScript de Facebook para construir interfaces de usuario, llamada React, pero en lugar de apuntar al navegador, apunta a plataformas móviles.

Las aplicaciones móviles nativas son aplicaciones desarrolladas específicamente para cada sistema operativo (Android, iOS), adaptando a cada uno el lenguaje con el que se desarrolla: Java para Android y Objective-C para iOS.

Gracias a React Native, los desarrolladores web escriben aplicaciones móviles como si fueran nativas pero que son compatibles con ambos sistemas operativos y son programadas en JavaScript.

2.7 Expo

Expo [8] es una plataforma y marco universal para aplicaciones React. Consiste en un conjunto de herramientas y servicios creados en torno a React Native y plataformas nativas que lo ayudan a desarrollar, construir, implementar e iterar rápidamente en iOS, Android y aplicaciones web desde la misma base de código JavaScript.

Se está convirtiendo en un ecosistema propio y consta de cinco herramientas interconectadas: XDE, Expo CLI, Expo Client, Expo Snack y Expo SDK. Entre ellas cabe destacar el uso de Expo Client, una aplicación móvil para Android e iOS que permite ejecutar proyectos React Native. Esto permite a los desarrolladores recargar en caliente en un dispositivo real o compartir el código de desarrollo con cualquier persona.

Las ventajas que destacan en Expo son que es gratuito y que facilitan la ejecución de aplicaciones móviles en desarrollo sin necesidad de instalarlas.

2.8 MongoDB

MongoDB [9] es un sistema multiplataforma de base de datos NoSQL programado en C++, orientado a documentos y de código abierto. MongoDB guarda estructuras de datos BSON, que es una especificación similar a JSON, con un esquema dinámico, haciendo de la integración de los datos en las aplicaciones un proceso más fácil y rápido.

Una de las capacidades de MongoDB es realizar consultas utilizando JavaScript, haciendo que estas sean enviadas directamente a la base de datos para ser ejecutadas.

La interfaz gráfica de usuario para tratar las bases de datos de MongoDB se llama MongoDB Compass y permite analizar y entender el contenido de sus datos sin un conocimiento formal sobre la sintaxis de consultas de MongoDB.

La elección de MongoDB se debe a su gran escalabilidad y su carácter descentralizado que hacen que soporten estructuras distribuidas. Las bases de datos NoSQL no requieren potentes recursos para poder trabajar con ellas. Además, ofrece más flexibilidad y permiten construir sistemas más abiertos que las bases de datos relacionales gracias a su fácil adaptación a nuevas evoluciones de las aplicaciones web. Siendo, gracias a MongoDB Compass, de fácil y simple uso.

2.9 Trello

Trello [10] es un software de administración de proyectos con interfaz web y disponible para iOS y Android. Emplea el sistema Kanban y el registro de actividades, que se realiza mediante tarjetas virtuales, permite organizar tareas, agregar listas, adjuntar archivos, etiquetar eventos, agregar comentarios y compartir tableros.

Trello ha sido la herramienta utilizada para realizar un seguimiento de la evolución del proyecto y organizar la planificación del mismo.

Entre sus ventajas destaca que es un software gratuito y con un interfaz simple y de sencillo uso.

3. ESTADO DE LA CUESTIÓN

En este capítulo se describe el panorama actual en relación con el tema que trata el proyecto. Para ello es necesario hablar sobre el contexto tecnológico en que se encuentra este y realizar una investigación de antecedentes y tecnologías similares.

3.1 Contexto tecnológico

La autenticación de dos factores es una manera de permitir al usuario acceder a un servicio por medio de una combinación de dos métodos de autenticación, duplicando, en consecuencia, la seguridad.

La segunda medida de autenticación puede ser un código PIN, una contraseña adicional, una contraseña de uso único (OTP), una huella dactilar o un token instalado dentro del dispositivo del usuario. Como vemos, la variedad de opciones con las que implementar A2F es muy amplia. Estas se dividen en factores de conocimiento (algo que se sabe), factores físicos (algo que se posee), y factores inherentes (algo que se es).

La autenticación es popularmente realizada mediante la utilización de factores de conocimiento. Para ello, el usuario debe memorizar y no compartir con nadie una contraseña, que puede ser más larga e incluyendo múltiples palabras o más corta como un pin alfanumérico.

Un caso específico de la autenticación con factor de conocimiento son las preguntas de seguridad. Se trata de un método poco seguro, ya que preguntas como "¿Cuál es tu colegio de la infancia?", suelen referirse a datos que probablemente son de dominio público, conocidos por el entorno del usuario, o fáciles de averiguar investigando la vida de este.

Por otro lado, los factores físicos han estado en uso desde que se tiene conocimiento. Se trata de un método análogo a la llave que abre una cerradura, en la que la llave simboliza un secreto que se comparte con la cerradura y este se guarda en sistemas computacionales que necesitan de la llave para revelarlo.

Los factores físicos pueden ser token de seguridad o cadenas de caracteres. Un token de seguridad es un dispositivo físico utilizado para acceder a un recurso que está restringido

electrónicamente. Actúa como una llave electrónica que contiene información secreta, que puede ser:

- Contraseña estática: la contraseña no es visible al poseedor y no varía, sino que se retransmite en cada autenticación por medio de un interfaz que puede ser USB, Bluetooth, comunicación de campo cercano (NFC) o identificación por radiofrecuencia (RFID).
- Contraseña dinámica síncrona: el verificador y el token están sincronizados y rotan la contraseña entre varias combinaciones que son producidas por un algoritmo criptográfico a la vez.
- Contraseña asíncrona: mediante una libreta de un solo uso, u otro algoritmo criptográfico, pero sin la intervención de un reloj, se crea una contraseña única y que sólo se utiliza una vez.
- Prueba de posesión de una clave privada: el verificador cifra con una clave pública un número aleatorio y el token contiene la clave privada necesaria para descifrarlo sin necesidad de revelar la clave.

Los tokens de seguridad se dividen en dos grupos: tokens conectados y tokens sin conexión.

Un token conectado es aquel dispositivo que transmite automáticamente la información ya que se conecta físicamente con el verificador. Estos tokens incluyen lectores de tarjetas, etiquetas inalámbricas o tokens USB.

Un token sin conexión no tiene contacto físico con el servidor de autenticación. Normalmente, este genera un código de autenticación que se introduce manualmente en el verificador. Un ejemplo de estos es el token SecurID de RSA que genera un código de seis dígitos que cambia cada 60 segundos a partir de una semilla pseudoaleatoria que es diferente para cada token fabricado.

Figura 3: Token SecurID RSA

Los tokens instantáneos también están disponibles y proporcionan lo que se considera el segundo factor de autenticación más utilizado, un código de token enviado por correo electrónico o vía SMS, eliminando la necesidad de proporcionar un token físico al usuario.

Por último, los factores inherentes o "algo que el usuario es" consisten en datos biométricos del usuario como pueden ser huellas digitales, iris, reconocimiento de voz, etc.

Aunque el segundo factor de autenticación sea uno de los mencionados anteriormente, el factor de autenticación que le precede, y la primera barrera de seguridad suele siempre ser una contraseña.

Incluso Google aseg ura en su web que el robo de las contraseñas es más fácil de lo que imaginamos. Además, es muy probable que los usuarios utilicen la misma contraseña para diferentes servicios, por lo que se facilita el acceso a todos ellos. Por ello, plataformas digitales como Whatsapp, Google o Instagram cuentan ya con la autenticación de doble factor.

El aumento del 60% en el robo de contraseñas de 2018 a 2019 quiere decir que ahora más que nunca es necesario aplicar la máxima seguridad a nuestra información online. Destaca el caso de Collection #1, un conjunto de más de 773 millones de direcciones de correo electrónico únicas y 21 millones de contraseñas únicas que se filtraron en la internet

oscura en enero de 2019. Violaciones de datos como esta son cada vez más comunes y peligrosas en Internet.

La autenticación de dos factores gana aún más relevancia hoy en día, debido a la transformación global que está teniendo lugar por la pandemia del COVID-19. La situación actual, ha llevado a la digitalización de millones de tareas que, por seguridad de los trabajadores, se están realizando por teletrabajo. Son muchos quienes piensan que la metodología de trabajo quedará afectada para siempre y que a partir de ahora incrementará mucho esta forma de trabajo que hasta antes de la pandemia sólo realizaban un 4’8% de ocupados en España [11].

Conferencias por aplicaciones como Zoom, reuniones a distancia por Skype o Jitsi Meet, o clases por videollamada desde Blackboard Collaborate o BigBlueButton son algunas de las tecnologías cuyo uso ha crecido exponencialmente con motivo de la pandemia. Con ello también ha incrementado la actividad de los hackers, que aprovechan que hoy más que nunca la información se moviliza de forma telemática.

3.2 Motivación

Actualmente, aunque pocos estarían en desacuerdo sobre la importancia de nuestras identidades digitales hoy en día, el uso de la autenticación de dos factores sigue siendo menor del que debería.

Un número sorprendentemente pequeño de usuarios hace uso de dos factores de autenticación, que es una capa de defensa inmediata para nuestras cuentas en línea a través de la autenticación secundaria del usuario. Si las identidades en línea del usuario particular no están debidamente protegidas, no sólo estaría en riesgo su información personal, sino que, si estos están relacionados con compañías o empresas más grandes, cualquier fallo de seguridad podría suponer daños que costaría millones de euros remediar.

La razón principal por la que tan sólo un número reducido de usuarios utilizan la autenticación de dos factores es que resulta un proceso tedioso. La autenticación de dos

factores que hace uso de códigos de acceso por medio de SMS al número de teléfono introducido previamente por el usuario, conlleva que, si el usuario pierde el teléfono, no lo lleva consigo en el momento en que quiere acceder a una cuenta o cambia de tarjeta SIM, por tanto, de número de teléfono, no podrá proceder y no será capaz de autenticarse.

Además, para los hackers no es difícil infiltrarse en el sistema SMS e interceptar los códigos o incluso utilizar la ingeniería social para asignar el número de la víctima a su propio número de teléfono tras llamar al proveedor móvil de la víctima. Sin embargo, este tipo de ataques son muy puntuales y dirigidos a personas concretas por razones específicas, no forma parte de los ataques más comunes.

En el campo laboral, las empresas utilizan llaves tokens de conexión en lugar de códigos de verificación, esto requiere que los empleados lleven el token encima en todo momento. Esto, aunque aporta bastante seguridad requiere hardware adicional, que hoy en día es algo que cada vez más tiende a reducir mediante la digitalización.

En el ámbito personal, la mayoría de usuarios consideran importante la seguridad de sus cuentas en un primer momento, por ejemplo, cuando crean un nuevo perfil o acceden a su banco online por primera vez. Sin embargo, a medida que pasa el tiempo, tienden a caer en el error de eliminar esta segunda capa de seguridad a cambio de ganar algo de tiempo en el proceso de autenticación y creyendo que nunca serán ellos las víctimas de un robo de contraseñas.

Por esto es tan importante encontrar un método de implementación del segundo factor de autenticación que sea amigable y atractivo para el usuario y que conlleve a un aumento considerable en la utilización de esta capa adicional de seguridad.

Por todo esto, la firma global de consultoría en desarrollo de productos, tecnología e innovación, Cambridge Consultants, fomentó el desarrollo del proyecto AudioMark, con el que pretende proporcionar la seguridad adicional de A2F creando un método fácil de usar a través de la aplicación de marcas de agua en audios. De esta manera, no será necesaria la espera y memorización de códigos para su posterior escritura como es el caso de los populares códigos de verificación recibidos por SMS.

Con AudioMark el usuario empleará en primer lugar su nombre de usuario y contraseña y a continuación, como segundo factor de autenticación, "algo que tiene" que será ese audio que contiene la marca de agua.

Con este proyecto se busca crear una metodología que combine la autenticación con los audios. La primera idea de caso de uso es en grandes empresas como por ejemplo T-Mobile, que puedan utilizar el tono comercial de la empresa como audio en el que introducir una marca de agua. De esta manera, todos los empleados de la empresa utilizarían el mismo tono y además de aportar seguridad, aportaría a todos los trabajadores un sentimiento de pertenencia a la empresa por medio del uso de su tono.

3.3 Tecnologías similares

Tras investigar sobre proyectos similares a AudioMark podemos encontrar dos posibles tecnologías que proponen nuevas metodologías de segundo factor de autenticación utilizando también audios.

En primer lugar, SoundLogin propone mediante una aplicación móvil y una extensión que se añade al navegador de escritorio, convertir los códigos de acceso recibidos por SMS en sonidos. Estos sonidos son transferidos a través del altavoz del móvil al verificador, que debe tener un micrófono.

La lista de servicios compatibles con SoundLogin es muy amplia e incluye plataformas como Google, Facebook, AWS, y GitHub.

Sin embargo, el problema sigue estando en que, si el usuario no tiene acceso a ese SMS, el proceso de autenticación queda estancado.

Por otro lado, Sound-Proof basa la A2F en el ruido ambiental. No requiere interacción entre el usuario y su teléfono, ya que el método de autenticación que sucede a la contraseña es la proximidad del verificador (suele ser un PC) al móvil del usuario. Para comprobar dicha proximidad, Sound-Proof compara el ruido ambiental grabado por ambos dispositivos. La experiencia del usuario es muy similar a la de una autenticación únicamente por contraseña.

En resumen, aunque estos dos sistemas plantean conceptos similares a AudioMark, como veremos en el siguiente capítulo, su funcionamiento guarda grandes diferencias.

4. DEFINICIÓN DEL TRABAJO

En este capítulo se expone cómo ha sido la realización del proyecto, los objetivos, la planificación y metodología del mismo.

4.1 Objetivos

Como se menciona en el capítulo anterior, este proyecto se ha realizado por requerimiento de la empresa Cambridge Consultants. Viendo la necesidad existente en el mercado, consideraron que la utilización de audio para aportar el segundo factor de autenticación, sería apropiada para crear un método innovativo y sencillo de utilizar.

Por ello, los objetivos principales del proyecto son los siguientes:

1. Encontrar un método para insertar un código de verificación en un audio. Este deberá ser lo suficientemente robusto como para soportar el ruido ambiental o la reproducción del audio a cierta distancia del verificador.

2. Suministrar la seguridad propia de un método de autenticación, es decir, proteger el código que actúa como segundo factor de autenticación para que no pueda ser replicado.

3. Construcción del entorno tecnológico para llevar a cabo la verificación por audio en dos pasos.

4.2 Metodología

Al realizar este proyecto para un cliente, es fundamental un proceso que preceda al inicio de la realización de este para su correcta elaboración, la ingeniería de requisitos.

La ingeniería de requisitos es una de las partes cruciales en el éxito de todo proyecto software, porque permite definir con la mejor calidad posible las características de un sistema software que satisfaga las necesidades del cliente.

Para ello, los primeros pasos a dar en el inicio del proyecto es la entrevista con el cliente. En ella, los principales requisitos del proyecto son recogidos y es a partir de estos donde empieza a elaborarse el trabajo.

A partir de ahí, se ha elegido una metodología para la realización del proyecto. Las metodologías son utilizadas en el desarrollo de proyectos porque aportan funcionalidad y organización.

En el caso de este trabajo se ha elegido la metodología Ágil. Hoy en día es esta la metodología más apropiada para un proyecto software debido a su flexibilidad, ya que los requisitos y soluciones evolucionan con el tiempo según la necesidad de este. El desarrollo del proyecto se divide en Sprints, que son entregas parciales y regulares del producto final. De esta manera en el inicio de cada Sprint, que en el caso de este trabajo tenía una duración de un mes, se establecían los objetivos y tareas a realizar en ese periodo. La duración del proyecto ha sido de 6 meses, un total de 6 Sprints.

En Trello, se han ido añadiendo tres grupos de tarjetas por cada Sprint: “Sprint # To Do”, “Sprint # In Progress”, “Sprint # Complete”, realizando así un seguimiento de las tareas. Parte del resultado final de esas tarjetas se muestra en la siguiente ilustración.

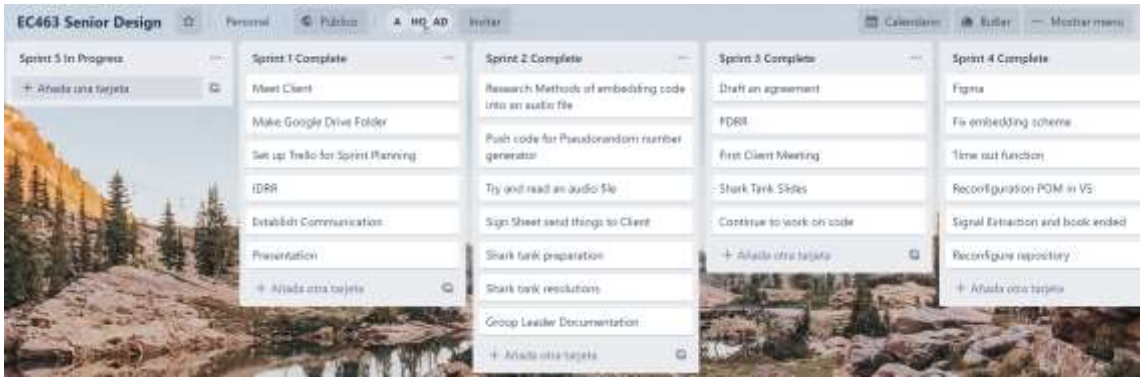


Figura 4: Trello con la planificación de tareas en tarjetas.

La planificación general del proyecto, con las distintas fechas de entrega de documentación y desarrollo y test del prototipo, se muestra en el diagrama de Gantt.

Figura 5: Diagrama de Gantt con la planificación de AudioMark.

4.3 Estimación económica

Otra consideración importante a la hora de realizar un trabajo de este tipo es la estimación de un presupuesto necesario para llevar a cabo su desarrollo.

Para ello, las consideraciones de los gastos deben ser realizadas detalladamente y definir cualquier posible gasto, para que no sea necesaria una ampliación presupuestaria posterior, que pueda conllevar a una deceleración en la evolución del trabajo.

Al ser AudioMark un proyecto software en su totalidad, la mayoría de herramientas tecnológicas, definidas con anterioridad, son gratuitas. Sin embargo, para desarrollar ese software se requiere la utilización de diferentes elementos hardware.

Los diferentes costes de los elementos que se emplearon para la realización del proyecto se exponen en la siguiente tabla.

Elemento	Descripción	Coste
Teléfono móvil	iPhone 8 con 64GB de almacenamiento	799,99€
Ordenador	Portátil con 512GB de disco duro, 16GB de RAM y procesador i7.	2.124,45€
AWS S3	Herramienta descrita en el capítulo 2: Descripción de las Tecnologías	Gratuito durante el primer año. Después: - Hasta 50TB/mes: 0,021€/GB - Siguientes 450TB/mes: 0,020€/mes - Más de 500TB/mes: 0,019€/mes
Otras herramientas	Demás herramientas descritas en la descripción de las tecnologías	0€
Trabajadores	Coste del trabajo necesario para la realización del proyecto	55€/hora: 33.000€
Total		**35.924,44€**

Tabla 1: Estimación económica del proyecto

La estimación económica calculada en la tabla anterior considera los gastos actuales, teniendo en cuenta que es el primer año de utilización del almacenamiento de Amazon Simple Storage Service, por tanto, no supone un gasto adicional. Esto cambiará cuando expire el periodo gratuito al cabo de un año desde que comenzó a utilizarse. Aunque el aumento en el presupuesto no será significativo, es importante tenerlo en cuenta para futuros desarrollos del trabajo.

5. SISTEMA/MODELO DESARROLLADO

En este capítulo se expondrá detalladamente en qué consiste AudioMark y de qué manera satisface los objetivos citados en el capítulo anterior. Al ser un trabajo complejo, antes de profundizar en el sistema, es necesario comprender un enfoque más general. De esta manera, se entenderán las diferentes partes del trabajo y cómo se relacionan entre sí antes de estudiarlas en detalle.

5.1 Idea y estructura general

AudioMark es un proyecto que consiste en la inserción de un código de verificación, de la misma tipología que los que se envían por SMS, en un audio. De esta forma el usuario no tendrá que insertar manualmente el código, tan solo tendrá que reproducir el audio frente al verificador.

Es necesario comprender tanto la estructura hardware de AudioMark como su software. Para ello, se muestra en la siguiente figura el diagrama hardware del trabajo.

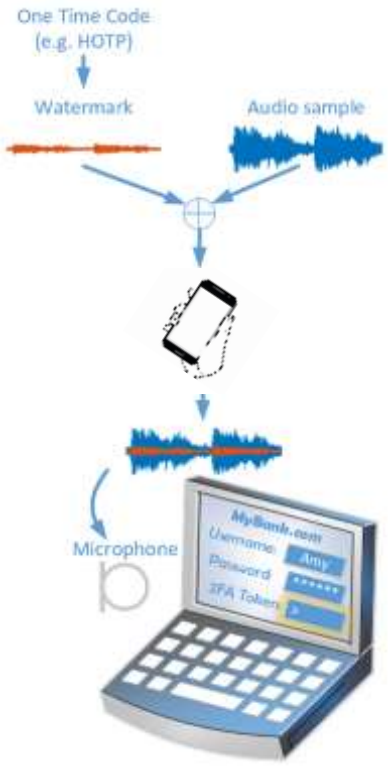


Figura 6: Diagrama de bloques del sistema

Para un conocimiento más profundo del funcionamiento de AudioMark en la siguiente ilustración se muestran sus diferentes componentes y cómo se relacionan entre sí.

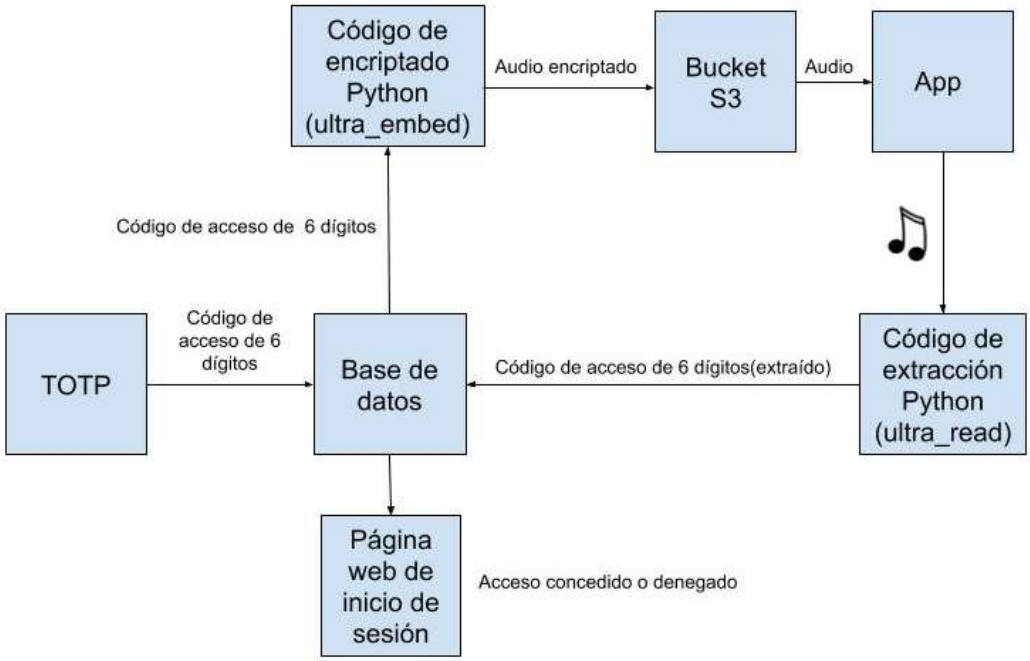


Figura 7: Diagrama de bloques del sistema

El producto final consistirá en una aplicación móvil multiplataforma (llamada App en el diagrama de bloques) que será desde donde se reproduzca el audio, una página web para la autenticación, un servidor que gestiona la autenticación y una base de datos para almacenar la información de inicio de sesión (nombre de usuario, contraseña, código de acceso, código recibido por audio).

El código que está formado por 6 dígitos, será generado por el algoritmo de autenticación TOTP y cambiará cada 30 segundos. Una vez se haya generado el código, este se almacenará en la base de datos.

Para encriptar el audio, el programa que embebe los seis dígitos en el audio tendrá que acceder a la base de datos para conseguirlos y proceder a su inserción en el sonido elegido, que para la puesta en marcha de este trabajo ha sido el tono de la compañía telefónica T-Mobile. El encriptado del audio a modo de marca de agua, que también denominaremos *audiomarking* se realizará de manera que el código no sea reconocible.

Además, el método de inserción del código tendrá que soportar modificaciones debidas al ruido en la transmisión. Por tanto, gran parte de este proyecto se dedica a investigar diferentes métodos, probarlos y decidir si son válidos o no, hasta encontrar uno que lo sea y poder aplicarlo.

Una vez tengamos el audio con el código encriptado, este se almacenará en un *bucket* de Amazon S3, para poder acceder a él desde la aplicación móvil y reproducirlo en la página web de autenticación una vez hayamos insertado usuario y contraseña. El verificador extraerá el código que reciba en el audio y lo almacenará en la base de datos para posteriormente realizar la comparación entre los dígitos recién extraídos y el código generado por TOTP que también se almacenó en la base de datos inicialmente.

Si tanto el nombre usuario, la contraseña y el segundo factor de autenticación coinciden se concederá acceso al usuario y se le denegará en caso contrario.

5.2 Generación del código de autenticación

Existen diferentes algoritmos de autenticación, como HOTP o TOTP. HOTP [12] es un algoritmo de contraseña de un solo uso (*One-Time Password*, OTP) basado en HMAC (código de autenticación de mensajes en clave-hash). OTP es una de las formas más sencillas y populares de códigos de verificación utilizados como segundo factor de autenticación.

El algoritmo HOTP se basa en un contador ascendente y una clave estática y simétrica sólo conocida por el token y por el verificador. Para crear un valor haciendo uso de HOTP se utiliza por defecto HMAC-SHA-1. Este podría ser reemplazado por las funciones hash HMAC-SHA-256 o HMAC-SHA-512. El valor HOTP se consigue a partir de truncar la salida de HMAC-SHA-1 para convertirla en datos útiles.

Por otro lado, TOTP [13] extiende HOTP y genera una contraseña de un solo uso, pero en lugar de utilizar un contador ascendente como semilla para generar el código, utiliza la unicidad del tiempo actual. Es decir, tomando el valor del tiempo en formato Unix, TOTP genera un código único de autenticación. Para realizar un correcto proceso de

autenticación, los relojes de ambas partes, probador y verificador deben estar estrictamente sincronizados.

El valor TOTP se calcula siguiendo la siguiente fórmula: Valor TOTP(K) = Valor HOTP (K, C_T), siendo K una clave secreta y C_T el valor que cuenta en número entero la diferencia entre el tiempo actual y la medianoche del 1 de enero de 1970 en horario UTC.

Debido a la latencia tanto humana como de las redes computacionales, y al mínimo asincronismo entre relojes, la OTP varía cada cierto intervalo de tiempo, para que dé tiempo a que se produzca la validación entre ambas partes. En el caso de AudioMark, el código de seis dígitos cambia cada 30 segundos, que es el mínimo tiempo necesario para ejecutar todo el proceso de autenticación sin que cambie el código y falle el proceso.

Es importante que ese código dependa del tiempo porque de esta manera, aunque un atacante consiga averiguar el código, cosa que como veremos más adelante será muy difícil, este no será replicable, porque las posibilidades de que el atacante descubra el código (además tendría que tener la contraseña del usuario-víctima) y lo utilice antes de que pasen 30 segundos son mínimas.

5.3 Estructura de la base de datos

Una vez se genera el código de seis dígitos, se almacenará en la base de datos. Para mostrar la estructura de la base de datos de AudioMark se hace uso de MongoDB Compass. Para ello, debemos abrir el interfaz gráfico y utilizar la siguiente cadena de conexión para acceder al clúster donde alojaremos nuestra base de datos: mongodb+srv://Admin:admin@cluster0-yrpqd.mongodb.net/test?authSource=admin&replicaSet=Cluster0-shard-0&readPreference=primary&appname=MongoDB%20Compass%20Community&ssl=true.

En MongoDB, al igual que en otras bases de datos que almacenan documentos BSON, estos se almacenan en colecciones [14]. BSON es un formato similar a JSON pero que se

representa de manera binaria. Al igual que JSON, es un formato para el intercambio de datos entre un servidor web y una aplicación web. La ventaja de BSON es que extiende JSON para que incluya tipos de datos como *UTC Datetime* para fechas o *Timestamp* para representar los segundos pasados desde el 1 de enero de 1970 [15].

Por defecto existen en el clúster al que nos hemos conectado las bases de datos "admin", "config" y "local", y aunque estas dos últimas tienen una y seis colecciones respectivamente, estas están vacías. La base de datos que utilizaremos para AudioMark se llama "login". Dentro de esta tenemos la colección "users" en la que introduciremos un documento por cada usuario. Para el desarrollo del proyecto se crearon dos usuarios y se utilizó cada uno de ellos para probar diferentes componentes del proyecto. Sin embargo, finalmente todas se reunieron en el usuario con nombre *user1*.

En la siguiente ilustración se muestran las bases de datos del clúster, y se puede comprobar que la única que almacena datos en ella es "login", con un tamaño de 36.9KB.

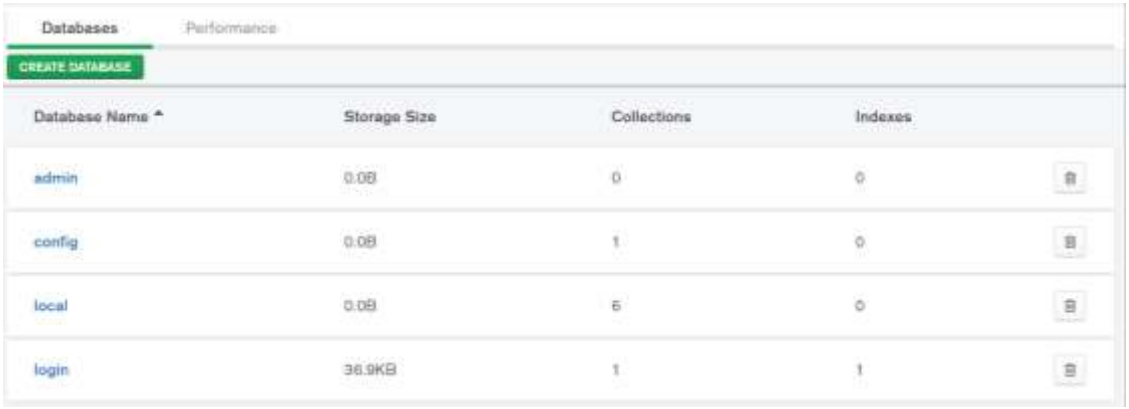


Figura 8: Clúster con las bases de datos de MongoDB

Si entramos en "login", podemos ver la colección "users" con los dos documentos, uno para cada usuario mencionado.

Figura 9: Colección de la base de datos "login"

En la siguiente figura se muestran los dos documentos de "users". Ambos tienen la misma estructura, aunque no tendrían por qué ya que al ser colecciones de documentos cada uno puede tener una estructura independiente, no como en bases de datos SQL.

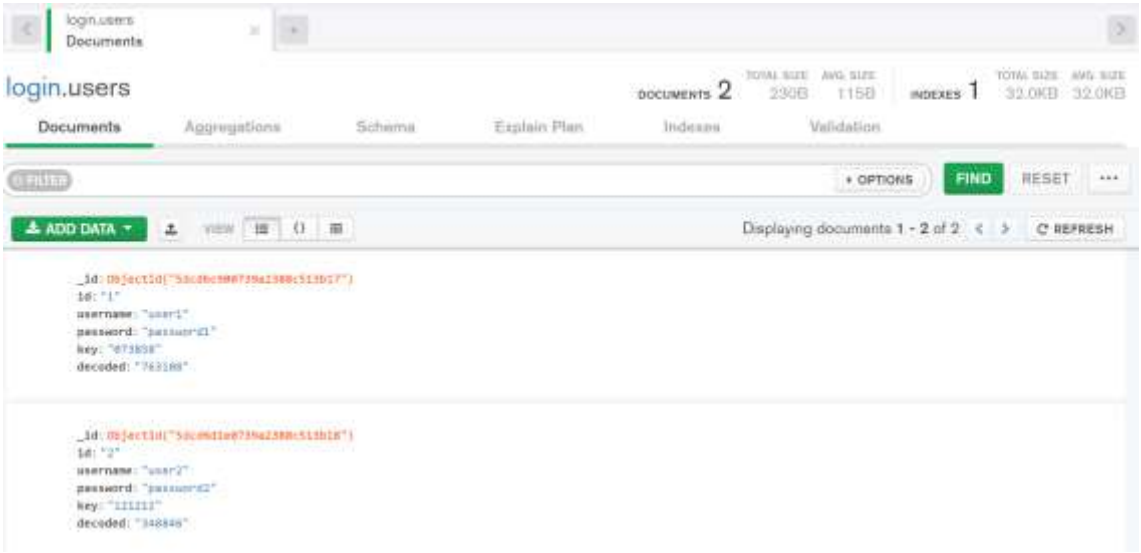


Figura 10: Documentos de la colección "users"

Además de un ID de objeto que tienen por defecto, estos documentos, que representan la información de cada usuario, tienen un ID (*id*), un nombre de usuario (*username*), una contraseña (*password*), una clave de acceso (*key*) y una clave descifrada (*decoded*).

Como vemos, tanto *key* como *decoded* son códigos de seis dígitos. *Key* representa el código de autenticación generado por TOTP que cambia cada 30 segundos y *decoded* es el código que como veremos más adelante se extrae del audio. Para conceder o denegar acceso a la sesión del usuario, el verificador comprobará el segundo factor de autenticación comparando estos dos campos de información.

5.4 Encriptado

Una de las partes más importantes de AudioMark, si no la que más, ha sido el proceso de encontrar un método de encriptado con el que introducir el código de seis dígitos en el tono elegido.

Para ello, se ha realizado un proceso de investigación y prueba y error que ha durado varios meses, pues cada vez que se trabajaba con un método se pasaba por las siguientes fases:

1. Investigación teórica sobre el método: para considerar si podría satisfacer los requisitos del proyecto.
2. Elaboración de código que encripte implementando ese método: se crea un programa que encripte el audio utilizando dicho procedimiento.
3. Elaboración de código que extraiga el código implementando ese método: se realiza el proceso contrario, que normalmente es invertir el proceso de encriptado, creando un programa que obtenga el código encriptado utilizando dicho procedimiento.
4. Prueba estática: en un mismo sistema, se prueba la correcta inserción y extracción del código.
5. Puesta en práctica del proceso: se realiza la prueba dinámica del procedimiento, es decir, se encripta el audio en un sistema A, se transmite mediante la reproducción del sonido con el código encriptado a un sistema B y se extrae el código recibido en el sistema B.

Este proceso se reiteró múltiples veces hasta que se ha adoptado un método que cumple con todos los requisitos y objetivos de AudioMark en todas estas fases.

Antes de encontrar el método definitivo y que es el que finalmente implementa AudioMark en el encriptado del código se han probado diferentes procedimientos. Además, los primeros métodos se desarrollan con MATLAB, pues favorece la representación y manejo de audio, pero posteriormente se cambia al lenguaje Python y el IDE Visual Studio Code para facilitar la integración con los demás componentes software.

A continuación, se exponen los diferentes métodos analizados.

5.4.1 Modificación de la amplitud de la señal de audio

Es importante destacar que, al empezar el desarrollo del encriptado del audio, se consideraba que el método elegido sería el utilizado en el producto final. Sin embargo, como es propio de un proyecto software de esta envergadura, a medida que ha avanzado el trabajo han ido apareciendo problemas que han conllevado a tener que hacer modificaciones como cambiar de forma de alterar la señal para insertar el código.

Inicialmente, se optó por utilizar MATLAB para modificar la amplitud de la señal de acuerdo con los 6 valores de cada dígito del código de acceso. Se optó por crear un vector de ceros de la misma longitud que la información de la señal de audio, que había sido leída con la función de la caja de herramientas de MATLAB Audio Toolbox™, `audioread`. Después, utilizando un bucle `for` se insertarían uno a uno los dígitos del código separados uniformemente por una distancia igual a la sexta parte de la longitud de la señal de audio.

En caso de que el tono fuese estéreo, previamente era necesario convertirlo en mono para poder realizar estas operaciones con el audio.

Además, como todos los componentes comenzaron a elaborarse por separado, el código de 6 dígitos era introducido manualmente, en lugar de ser generado por TOTP.

Una vez el vector de ceros incluía los dígitos, este se sumaba a la señal de audio, que como hemos mencionado también era un vector, y ambos eran de la misma longitud.

```
[y, Fsy] = audioread('jingle.mp3');
if(ismatrix(y)) % In case the audio jingle is stereo, we choose one of the two columns
    y=y(:,1);
end
%sound(y, Fsy);
t = 0:1/Fsy:(length(y)-1)/Fsy;
%plot(t,y);
code = [1 2 3 4 5 6];
interval = length(y)/6;
z = zeros(1, length(y));
z = z';
for i=1:6
    z(i*interval) = code(i);
end

final = z+y;
figure;
plot(t,final);
title('Jingle + Code');
xlabel('Time (s)');
ylabel('Amplitude');
sound(final, Fsy);
```

Figura 11: Código de encriptado por modificación de la amplitud del audio

La representación en el dominio del tiempo del resultado de la suma de la señal de audio y el código se muestra en la siguiente figura.

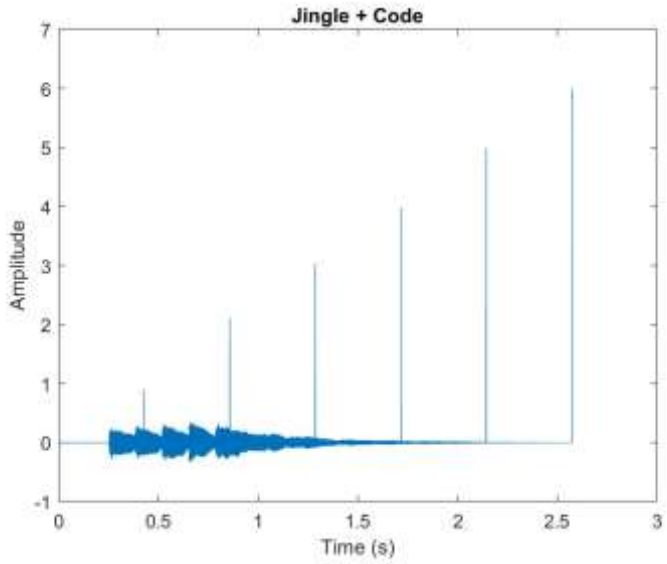


Figura 12: Representación del audio encriptado en dominio temporal

Aunque viendo la representación del audio podemos claramente saber que el código era [1 2 3 4 5 6], al escuchar este audio no se aprecia diferencia alguna con el tono original.

Este era un objetivo a conseguir pues si el tono con el código sonaba prácticamente igual que el sonido encriptado, sería más difícil para un atacante que no conozca el procedimiento seguido para insertar el código, llegar a extraerlo.

Como es lógico, el código de extracción realiza el proceso inverso al de encriptado y una vez recibía el audio encriptado, buscaba los valores de la señal en cada intervalo igual a la sexta parte de la longitud de la señal y obtenía uno a uno los dígitos del código.

Este método fue satisfactorio sobre todo por el hecho de que la alteración de la señal no era audible y el código extraído era el correcto. Sin embargo, pasada la fase 4 de la prueba estática, se procedía a grabar el audio el un sistema diferente al que lo generaba y esto introducía ruido.

Además, para que el sistema B, que actúa como verificador, inicie a grabar el audio en el mismo momento en que se comienza a reproducir el audio en el sistema A, pues ambos audios debían ser exactamente iguales para que el método funcionara, habría que añadir algo más al procedimiento.

Todo esto, sumado al hecho de que era un método bastante al descubierto y no aportaba la suficiente seguridad, se demostraba que el método no era lo suficientemente robusto como para ser utilizado en el proyecto.

5.4.2 Modificación de la frecuencia de la señal de audio

Partiendo del método anterior se intentó encontrar un método que solucionara el problema de la falta de seguridad por la sobreexposición del método de encriptado. Por esto, se optó por alterar la frecuencia de la señal e introducir en esta los dígitos del código.

Sin embargo, la modificación del espectro produciría grandes cambios en el sonido (a no ser que las modificaciones fueran en altas frecuencias), y su implementación sería más compleja.

5.4.3 LSB

LSB o Least Significant Bit hace referencia al bit de menor importancia de una serie de números en binario. En una arquitectura "little-endian" el valor menos significativo es almacenado en la posición de memoria más baja.

Para comprender en qué consiste que un bit sea más o menos significativo se puede utilizar la analogía de una cuenta bancaria, en la que el dinero sea la información y cada dígito del saldo sean los bits. Supongamos que tenemos una cuenta bancaria con 7962€ y tenemos que reducir en una unidad uno de los dígitos. Si reducimos el dígito más a la izquierda, pasaríamos de tener 7962€ a 6962€, lo que supone una pérdida de 1000€. Si redujésemos el segundo dígito por la izquierda pasaríamos de tener 7962€ a 7862€, con una pérdida de 100€. Sin embargo, si redujésemos el dígito más a la derecha, pasaríamos de tener 7962€ a 7961€, una pérdida casi inapreciable con respecto a la cantidad total.

Esto mismo ocurre con las pérdidas de información al modificar uno u otro bit. Si modificamos el bit menos significativo de un byte, la pérdida de información será inapreciable. Por ello, entre las técnicas digitales de la esteganografía destaca el uso de esta la inserción de información en el bit menos significativo. Su uso más común es en imágenes, aunque como veremos a continuación, también se puede aplicar en audios.

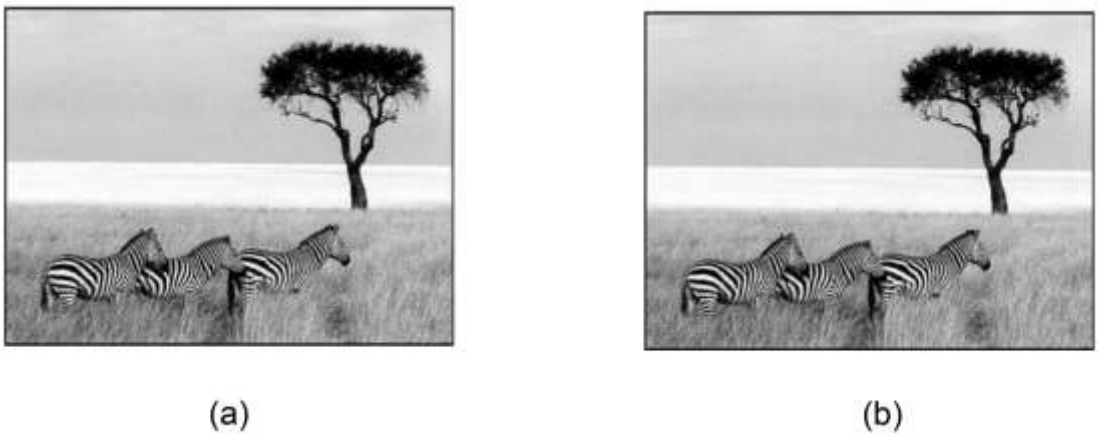


Figura 13: Ejemplo de LSB en una imagen
(a) Imagen de tres cebras en la sabana
(b) Imagen de tres cebras en la sabana y el texto completo de cinco obras de Shakespeare

La técnica consiste en reemplazar el bit menos significativo de un byte para introducir un bit de la información que se desea ocultar. Para posteriormente, extraer el bit menos significativo de los bytes recibidos y recomponer la información.

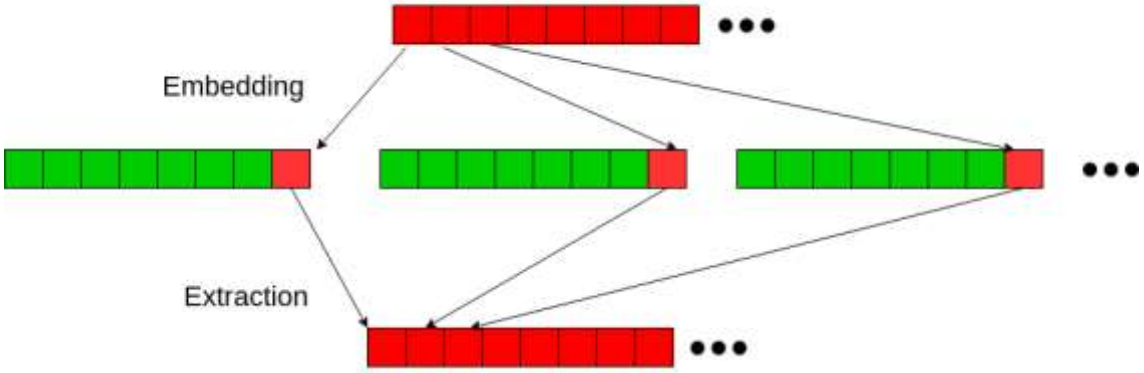


Figura 14: Diagrama de encriptado y extracción por LSB

La única desventaja es que el tamaño de la imagen o audio portador de la información oculta deberá ser mucho mayor que esta, pues por cada byte del portador sólo se podrá ocultar un bit. Es decir, un archivo podrá ocultar como máximo un mensaje del 12,5% de su tamaño.

La información a ocultar en AudioMark se trataba de 6 dígitos, cada dígito son 8 bits (un byte), un total de 48 bits. Por tanto, el audio debería ser de un mínimo de 48 bytes. El tono de T-Mobile tiene un tamaño de 41422 bytes, mucho más que suficiente para almacenar el código de acceso.

Después de haber investigado este método, y haber comprobado que podría llegar a ser satisfactorio para su utilización en el proyecto, se procedió a realizar el código que lo implementara. Esto se realizó ya en Visual Studio Code en el lenguaje Python.

```
# We will use wave package available in native Python installation to read and write .wav audio file
import wave
# read wave audio file
song = wave.open("jingle.wav", mode='rb')
# Read frames and convert to byte array
frame_bytes = bytearray(list(song.readframes(song.getnframes())))
print('Frame bytes length: ' + str(len(frame_bytes)))
bytes_len = len(frame_bytes)

# The "secret" text message
watermark='123456'

# Convert text to bit array
bits = list(map(int, ''.join([bin(ord(i)).lstrip('0b').rjust(8,'0') for i in watermark])))
print(bits)
print(len(bits)) #watermark length in bits is 48 because every digit is an ASCII (8 bits = 1 byte)
print(int(10/3))
# Replace LSB of each byte of the audio data by one bit from the text bit array
for i, bit in enumerate(bits):
    print(i)
    print(bit)
    frame_bytes[i] = (frame_bytes[i] & 254) | bit
    i = i + int(bytes_len/len(bits))
# Get the modified bytes
frame_modified = bytes(frame_bytes)

# Write bytes to a new wave audio file
with wave.open('song_embedded.wav', 'wb') as fd:
    fd.setparams(song.getparams())
    fd.writeframes(frame_modified)
song.close()
```

Figura 15: Código de encriptado LSB

Como vemos en la figura, el código de acceso (123456) aún era introducido manualmente, pues antes de conectar los distintos componentes se probaban por separado.

La sustitución del bit menos significativo se realiza mediante las operaciones lógicas AND (&) y OR (|). Para poner a cero el último bit de un byte, este se multiplica por 254 en binario, 11111110. A continuación, se suma el valor del bit que queremos ocultar, mediante el operador OR.

El nuevo audio con el código encriptado mediante la inserción de los 6 dígitos con LSB se guarda como *song_embedded.wav*.

```
# Use wave package (native to Python) for reading the received audio file
import wave
song = wave.open("song_embedded.wav", mode='rb')
# Convert audio to byte array
frame_bytes = bytearray(list(song.readframes(song.getnframes())))
bytes_len = len(frame_bytes)

# Extract the LSB of each byte
extracted = []
for i in range(0, bytes_len, int(bytes_len/48)):
    extracted.append(frame_bytes[i] & 1)
# Convert byte array back to string
string = "".join(chr(int("".join(map(str,extracted[i:i+8])),2)) for i in range(0,len(extracted),8))

# Print the extracted text
print("Sucessfully decoded: " + string)
song.close()
```

Figura 16: Código de extracción LSB

Este método cumplía con el objetivo de hacer un audio que contuviera el código sin modificar su sonido. Sin embargo, la modificación de un bit de cada byte no era lo suficientemente robusta como para llegar al sistema receptor exactamente igual que como salía del reproductor de audio. Una mínima diferencia impedía la correcta extracción del código. Por eso, este método también tuvo que ser descartado.

5.4.4 Ultrasonido Chirp SDK

Finalmente, ante las diferentes pruebas y error de los métodos anteriores, ha primado la necesidad de que el método de encriptado facilite al receptor cuándo comenzar a grabar el audio para extraer el código y a su vez que este no se pierda durante la transmisión.

Se ha recurrido al ultrasonido para insertar el código. Por ello, se han investigado técnicas que permitan la inserción de información en audios mediante ultrasonido.

De esta manera se ha llegado a Chirp [18], un kit de desarrollo de software (SDK) que permite a las aplicaciones enviar y recibir información usando sonido. Un “chirp” codifica un array de bytes como una señal de audio, en el caso de AudioMark ultrasonido para evitar que sea perceptible para el oído humano. Esta señal puede ser transmitida por cualquier dispositivo con un altavoz y recibida por cualquier dispositivo con micrófono

y Chirp SDK. Está diseñado para ser robusto a distancias de varios metros y en ambientes con ruido.

Para conseguir el objetivo de AudioMark, se utiliza el SDK de Chirp en la función `chirp` para convertir el código de seis dígitos en una señal de audio ultrasónica, llamada *signal.wav*. A continuación, esta se superpone al tono que se utiliza como portador y el audio resultante es el que se enviará al verificador.

```
def chirp(args):
    '''Function to create chirp Payload and save it as a wav file to be later used as a signal'''
    #INITALIZE CHIRP SDK
    sdk = ChirpSDK()

    #DISABLE AUDIO PLAYBACK
    sdk.audio = None
    sdk.start(send=True, receive=False)

    #ENCODE PAYLOAD
    id = args.unicode.encode('utf-8')
    payload = sdk.new_payload(id)

    #PROCESS OUTPUT
    output_file = "signal.wav"

    #SET UP ACTUAL SIGNAL
    w = wave.open(output_file, 'w')
    w.setnchannels(1)
    w.setsampwidth(2)
    w.setframerate(sdk.output_sample_rate)
    sdk.send(payload)

    #WRITE PAYLOAD TO AUDIO FILE
    while sdk.state == CHIRP_SDK_STATE_SENDING:
        data = ar.array('h', [0] * CHIRP_SDK_BUFFER_SIZE)
        byte_data = bytearray(data.tobytes() if sys.version[0] == '3' else data.tostring())
        sdk.process_shorts_output(byte_data)
        w.writeframes(byte_data)

    w.close()
    sdk.stop()
    return overlap()
```

Figura 17: Código de la función "chirp" que convierte los 6 dígitos en ultrasonido

En el espectro de la señal *signal.wav* se comprueba que todas sus componentes son de frecuencia 18 kHz a 20 kHz aproximadamente. Se considera que el oído humano tiene unos límites absolutos de audición de 20 Hz a 20 kHz, sin embargo, nuestra audición es más sensible en el rango de frecuencia de 2kHz a 5kHz [16]. Al escuchar *signal.wav*, antes de solaparla al tono de T-Mobile, no se percibe ningún sonido.

A continuación, con la función `overlap` se solapan el tono (tras un proceso de filtrado se denomina *f_jingle.wav*) y el ultrasonido (*signal.wav*) en un nuevo archivo .wav con el nombre *overlap.wav*, que será el que reproduzca nuestra aplicación móvil como veremos más adelante.

```
def overlap():
    '''Function to overlap signal and jingle'''
    #filter jingle
    lpf()
    sound1 = AudioSegment.from_file("f_jingle.wav")
    #sound1 = AudioSegment.from_file("f_jingle.wav")
    sound2 = AudioSegment.from_file("signal.wav")
    #ASSUMING SIGNAL IS LONGER THAN THE SOUND
    combined = sound2.overlay(sound1)
    combined.export("overlap.wav", format='wav')
```

Figura 18: Código de la función "overlap"

Antes de utilizar las funciones `chirp` y `overlap`, primero se accede a la base de datos para obtener el código de seis dígitos generado por TOTP, que está almacenado en el campo *key*.

```
if __name__ == '__main__':
    myclient = pymongo.MongoClient("mongodb+srv://Admin:admin@cluster0.ycpqd.mongodb.net/test")
    mydb = myclient["login"]
    mycol = mydb["users"]
    myquery = { "username": "user1" }
    mydoc = mycol.find_one(myquery)
    code = [mydoc["key"]]

    #print code
    print(code[0])
    realCode = code[0]

    parser = argparse.ArgumentParser(description = 'Chirp Audio Writer', epilog = 'Generates a .wav file containing a Chirp payload')
    parser.add_argument('-u', '--unicode', type=str, default=realCode, help='Unicode string used to generate payload')
    args = parser.parse_args()
    chirp(args)
```

Figura 19: Código "main" de encriptado

La ventaja de este kit de desarrollo software es que, además, incluye una serie de tonos en ultrasonido que permitirán al receptor, reconocer cuando comienza el ultrasonido que contiene los dígitos.

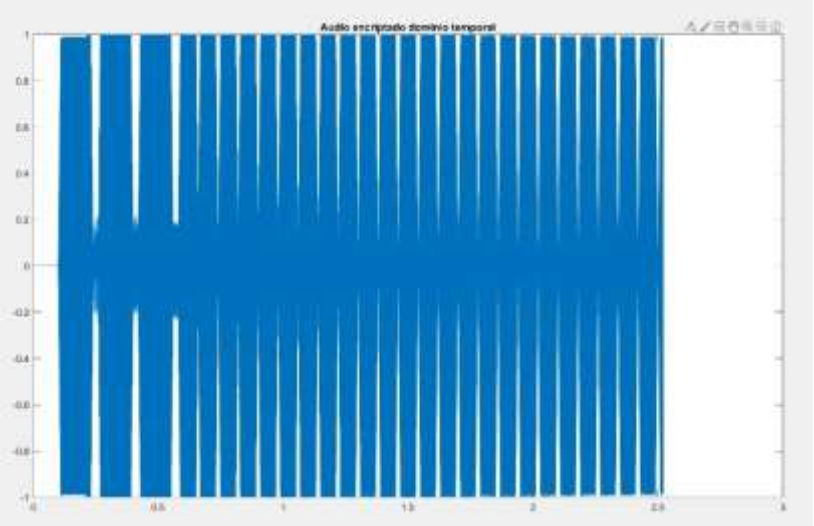


Figura 20: Dominio temporal del audio encriptado

Los primeros 0.56 segundos del audio encriptado contienen información que el receptor toma para saber que a continuación vendrá el código de acceso en ultrasonido. Como vemos en la figura 20, al solapar ambos audios (tono original y ultrasonido) en el dominio del tiempo afecta a ambos sonidos, por tanto, se percibe una leve modificación del tono original, pero cumple con las funcionalidades requeridas por el proyecto.

El código mostrado hasta ahora, que se encuentra en un archivo llamado *ultra_embed.py* es el que genera el audio encriptado, a continuación, se muestra el código que extrae el segundo factor de autenticación en el verificador, alojado en un archivo llamado *ultra_read.py*.

Este código contiene la función `on_received`, que extrae el ultrasonido, llamado *payload* y lo vuelve a convertir de sonido ultrasónico a código de seis dígitos. Una vez obtiene la clave, la escribe en la base de datos en el campo *decoded* y la muestra por la ventana de comandos.

```
def on_received(self, payload, channel):
    if payload is None:
        print('DECODE FAIL')
    else:
        #Access DB
        myclient = pymongo.MongoClient("mongodb+srv://Admin:admin@cluster0-yrpqd.mongodb.net/test")
        mydb = myclient["login"]
        mycol = mydb["users"]

        #set the key
        myquery = { "username": "user1"}
        decoded = payload.decode('utf-8')
        newvalues = {"$set": {"decoded":decoded}}
        mycol.update_one(myquery, newvalues)

        print('RECEIVED: {data} [ch{ch}]'.format(data=decoded, ch=channel))
        global loop
        loop = False
```

Figura 21: Código de la función "on_received"

Más adelante veremos como este archivo se ejecuta en el servidor antes de comparar que ambas claves coincidan.

Este ha sido finalmente el método seleccionado para el proyecto porque, tras probarlo en ambientes con todo tipo de ruidos, desde ruido blanco y gaussiano hasta ruido ultrasónico, se ha comprobado su eficacia. El código extraído siempre coincide con el encriptado.

Método de encriptado	**Fácil de implementar**	**Efectivo sin transmisión**	**Sincronización**	**Robusto**
Modificación amplitud	✔	✔	✘	✘
Modificación frecuencia	✘	✘	✘	✘
LSB	✔	✔	✘	✘
Chirp SDK Ultrasonido	✔	✔	✔	✔

Tabla 2: Comparación de los métodos de encriptado estudiados

5.5 Almacenamiento Amazon S3

Llegado este punto tenemos el audio encriptado, *overlap.wav*. El siguiente paso es conseguir una aplicación móvil que lo reproduzca. Sin embargo, este audio será dinámico, porque cambiará cada vez que cambie el código generado por TOTP, por ello se optó por alojar el audio en la nube y así, cada vez que cambie el audio, se actualizará el audio de la nube y la aplicación móvil reproducirá el audio correcto.

Para tener almacenamiento en la nube se ha elegido Amazon Simple Storage Service. Cada vez que se genere un nuevo audio, habrá que ejecutar el archivo Python que se encarga de subir el audio encriptado a la nube, llamado *upload.py*.

Para hacer uso de servicios como Amazon S3 o Amazon EC2 desde Python, se necesita el kit de desarrollo software de Amazon Web Services, llamado Boto3 [17]. Con este SDK, se accederá fácilmente al almacenamiento de Amazon S3.

```
import boto3

#change the bucket name 'audiomarkaws' to your s3 buckets name

s3 = boto3.client('s3')
s3.upload_file('overlap.wav','audiomarkbucket','overlap.wav')
```

Figura 22: Código de "upload.py"

En Amazon S3 los objetos se almacenan en contenedores, llamados buckets. Para el audio encriptado se ha creado un bucket con el nombre de *audiomarkbucket*. Tras ejecutar *upload.py*, se actualizará el audio en la nube y el aspecto del bucket será el que se muestra en la siguiente figura.

Figura 23: Bucket "audiomarkbucket"de Amazon S3

5.6 Aplicación móvil

Una vez el audio encriptado se encuentra en la nube, se necesita una aplicación móvil desde donde reproducirlo. Se optó por programarla con React Native Expo para que el mismo código fuese útil tanto para Android como para iOS y no fuese necesario publicar las aplicaciones en Play Store o App Store.

La aplicación permite al usuario iniciar sesión y reproducir el audio al verificador tras haber introducido el nombre de usuario y contraseña en este último.

Figura 24: Pantalla de inicio de sesión en la app

Se han añadido el nombre de usuario (*User1*) y contraseña (Test1) porque esto es un prototipo, en un futuro, esto no debería aparecer en la pantalla de inicio de sesión. Una vez hemos iniciado sesión, pasamos a la pantalla desde donde se reproduce el audio con un botón "Play/Pause".

Al pulsar este botón, la aplicación accede mediante el link: https://audiomarkbucket.s3.eu-west-3.amazonaws.com/overlap.wav a el audio del bucket

S3 y lo reproduce o lo pausa, depende de su estado anterior. Si estaba reproduciéndose, al pulsar el botón se detiene y viceversa.

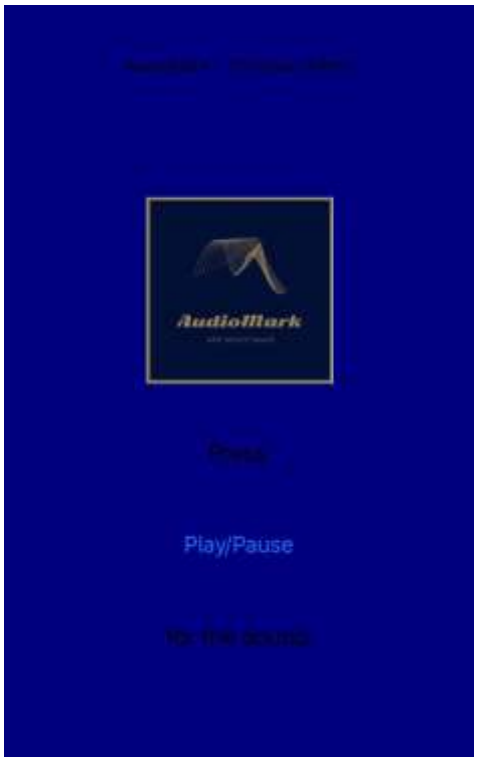


Figura 25: Pantalla de reproducción de audio en la app

5.7 Funcionamiento del servidor y la página web de autenticación

La página web está programada en Node.js y HTML. Consta del servidor, *webApp.js* y la página web de inicio de sesión, *index.html*. El servidor se encarga de varias cosas:

- Con la función `updateTotp` actualiza el valor del código de acceso cada 3 segundos en la base de datos, pero este sólo cambiará cada 30 segundos, que como está programado TOTP. Para esto, se ejecuta *sha_1.jar* como proceso hijo.

```
//Update TOTP every 3 seconds, and every 30 seconds it will change
function updateTotp(){

  let childProcess = require('child_process').spawn(
    'java', ['-jar', 'HMAC-SHA1/src/sha_1.jar']
  );

  childProcess.stdout.on('data', function(data) {
    var buf = Buffer.from(data);
    var keyToPutInDB = buf.toString();

    console.log(keyToPutInDB);

    MongoClient.connect(url, function(err, db) {
      if (err) throw err;
      var dbo = db.db("login");
      var myquery = { username: "user1" };
      var newvalues = { $set: {key: keyToPutInDB} };
      dbo.collection("users").updateOne(myquery, newvalues, function(err, res) {
        if (err) throw err;
        console.log("1 document updated");
        db.close();
      });
    });

});
}setInterval(updateTotp, 3000);
```

Figura 26: Código de la función "updateTotp"

- Con la función `runReadScript` ejecuta el archivo *ultra_read.py*: se encarga de extraer el código de acceso y muestra su salida por consola. Además, se ha añadido una función que muestre el segundo actual, ya que en el segundo 0 y segundo 30 de un minuto cambia el código TOTP. Para saber cuándo generar el audio con tiempo suficiente para reproducirlo antes de que cambie en la base de datos. Todo esto se realiza con un enfoque de probar el proyecto y su funcionalidad.

```
//Run the reading script on the server
function runReadScript(){
  return spawn('python', ["../python/ChirpSDK-Python-3.5.0 (1) 2/ultra_read.py"]);
}

const subprocess = runReadScript();

// print output of script
subprocess.stdout.on('data', (data) => {
  console.log(`data:${data}`);
  var second = (new Date()).getSeconds();
  console.log(second);
});
subprocess.stderr.on('data', (data) => {
  console.log(`error:${data}`);
});
subprocess.on('close', () => {
  console.log("Closed");
});
```

Figura 27: Código de la función "runReadScript"

- Verifica la autenticación: comprueba que el usuario y contraseña son los correctos y que la clave generada por TOTP y la clave extraída coinciden. En caso de pasar el proceso de autenticación satisfactoriamente, la sesión pasa a estar en estado *loggedin*.

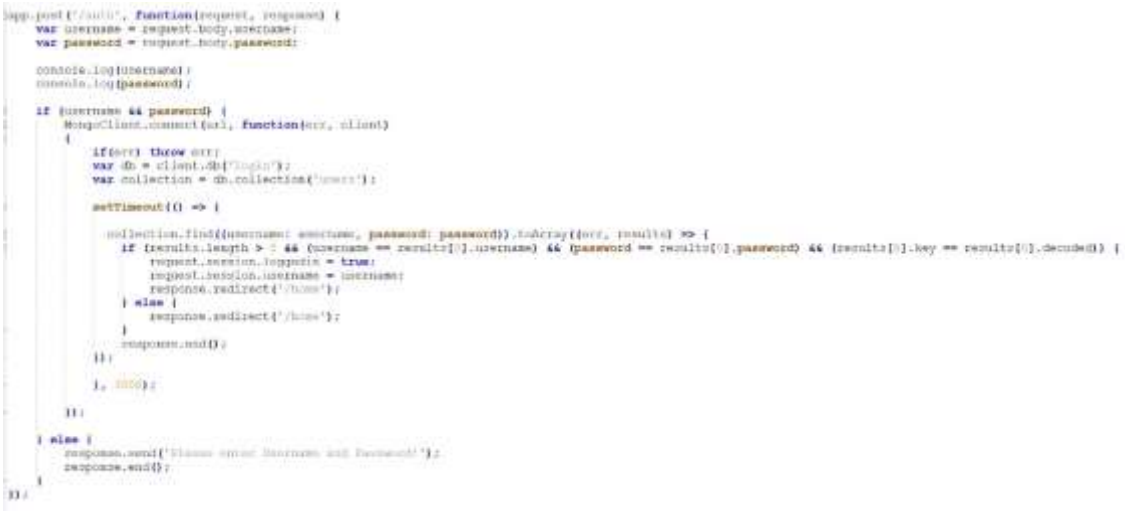


Figura 28: Código de verificación mediante el acceso a la BBDD

La página web inicial, *index.html*, consta de un formulario que recoge el nombre de usuario y contraseña y un botón de tipo *submit* que redirige a */auth*, donde el servidor verifica las entradas recibidas del formulario y decidirá si pasar a estado *loggedin* o no, como vemos en la figura 28. Es en esta página donde tras introducir nombre de usuario y contraseña, el usuario deberá reproducir el audio encriptado desde la aplicación móvil.

Es importante mencionar que mediante la directiva *listen* el servidor acepta peticiones entrantes solamente del puerto 8000. Por tanto, para acceder a este, se deberá ir a la siguiente URL: http://localhost:8000/.

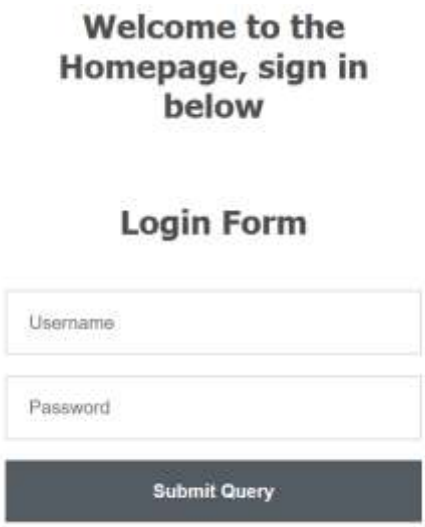


Figura 29: Página web verificadora

En caso de pasar el proceso de autenticación, se mostrará el siguiente mensaje en la pantalla: “Welcome back, user!”. En caso contrario, el mensaje será este: “Login Failed!”.

6. ANÁLISIS DE RESULTADOS

El proyecto AudioMark ha cumplido con todos los objetivos planteados inicialmente tanto funcionales como de seguridad, que están resumidos en el la **definición del trabajo**. Estos eran:

- Encontrar un método para insertar un código de verificación en un audio. Este deberá ser lo suficientemente robusto como para soportar el ruido ambiental o la reproducción del audio a cierta distancia del verificador.
- Suministrar la seguridad propia de un método de autenticación, es decir, proteger el código que actúa como segundo factor de autenticación para que no pueda ser replicado.
- Construcción del entorno tecnológico para llevar a cabo la verificación por audio en dos pasos.

Por lo tanto, tras pasar por diferentes procedimientos, se ha conseguido encontrar un método para encriptar un código en un audio que respetase la seguridad que requieren procesos de autenticación como este.

Además, al ser un método que hace uso de ultrasonido, la variación del sonido es mínima. Las grandes compañías que tienen tonos publicitarios y propios de su identidad como empresa podrán utilizarlos como audio portador en el segundo factor de autenticación de sus empleados.

En caso de usuarios particulares, podrán elegir un tono personalizado, ya que AudioMark permite utilizar cualquier archivo de audio como portador del código de acceso.

Para comprobar la robustez del método se descargó la aplicación móvil Noise Generator: Full Spectrum y se llevó a cabo el proceso de autenticación en entornos ruido blanco de frecuencia 20Hz a intensidad del 33%, 66% y 100%, y ruido ultrasónico de frecuencia 20kHz, 21kHz y 22kHz.

En la siguiente tabla se muestran los resultados del código extraído, siendo el código encriptado 831810:

Tipo de ruido	Nivel	Frecuencia	Código extraído
Sin ruido	0%	N/A	831810
Blanco	33%	20Hz	831810
Blanco	66%	20Hz	831810
Blanco	100%	20Hz	831810
Ultrasonido	N/A	20kHz	831810
Ultrasonido	N/A	21kHz	831810
Ultrasonido	N/A	22kHz	831810

Tabla 3: Análisis de comportamiento en entornos ruidosos

Como vemos en la tabla, la extracción del código es satisfactoria al 100%, para cualquier tipo de entorno ruidoso.

El requisito más importante del proyecto era la seguridad, pues es lo que un segundo factor de autenticación intenta garantizar en caso de que se dé un robo de contraseña. Esta característica se ha cumplido satisfactoriamente porque sin el método de extracción del código en forma de ultrasonido, aunque se consiga aislar el ultrasonido del audio encriptado, este no sería más que un sonido ultrasónico sin significado alguno.

En el caso extremo en que un atacante consiguiera transformar el ultrasonido en el código de seis dígitos utilizado como segundo factor de autenticación, seguiría siendo seguro porque cada 30 segundos ese código cambia, por tanto, no le serviría de nada ya que la probabilidad de que consiga extraerlo, descifrarlo y utilizarlo en menos de 30 segundos es mínima.

Por último, el entorno tecnológico utilizado consta de una aplicación móvil, que podría ser utilizada desde cualquier otro dispositivo como una Tablet u otro ordenador con

acceso a la aplicación, y de una página web que cumple con la función de simular un entorno de inicio sesión de lo que podría ser cualquier cuenta en una empresa, entidad bancaria o correo electrónico.

Esto soluciona los problemas que plantea que el segundo factor de autenticación sea enviado por SMS, ya que ante un cambio de móvil no se podría recibir ese código.

Por todo ello, el proyecto ha sido satisfactorio cumpliendo estos requisitos. Aunque para su aplicación en el mundo real habría que continuar con su desarrollo y realizar algunos cambios.

7. CONCLUSIONES Y TRABAJOS FUTUROS

Lo más importante de AudioMark es que llega a demostrar que existe una forma de autenticación de dos factores mediante el uso de audio como portador de la segunda capa de autenticación.

Para demostrarlo y hacer un prototipo útil, se desarrolla todo lo que envuelve AudioMark, incluyendo la aplicación móvil y la página de autenticación.

Sin embargo, trabajos futuros que se deberían realizar para implementar AudioMark en el mundo real incluyen:

- Adaptar el proyecto a multiusuario. Ahora mismo, la base de datos de MongoDB de AudioMark tiene dos únicos usuarios, pero sólo se hace uso de uno en todo el proceso. Inicialmente, uno de ellos se utilizaba para comprobar la modificación de la base de datos y otro para el proceso de autenticación. Sin embargo, una vez se comprobó que ambas funciones eran satisfactorias, se reunieron todas en uno de los usuarios para simular el proceso completo de autenticación. Si quisiéramos adaptar este proyecto a un ámbito más práctico, habría que posibilitar el añadido de usuarios y su incorporación a la base de datos para que cada uno tuviera su nombre de usuario, contraseña y código de acceso.
- Publicar la aplicación móvil en la tienda de aplicaciones como producto independiente y completado, en lugar de una aplicación en desarrollo como está en su estado actual. Esto incluirá que la aplicación se esté ejecutando continuamente, mientras que actualmente cada vez que queremos hacer uso de ella hay que ponerla en marcha desde el terminal, como se explica en el Anexo: Manual de Usuario.
- Mejora de la experiencia del usuario, de forma que el proceso tenga lugar de forma más automática, es decir, un desarrollo en la integración de los componentes.
- Aumento de funcionalidades como la selección del tono portador o la posibilidad de aumentar o disminuir el tiempo que tarde el código TOTP en cambiar.

- Aplicación de AudioMark en sistemas reales en los que probar su adaptación y nuevas necesidades del entorno práctico.

Estas son algunas de las futuras mejoras que se podrían aplicar a AudioMark para convertirlo en un proyecto real y útil para la autenticación segura en cuentas de los usuarios. Aunque los objetivos del proyecto actual se han cumplido, se podría mejorar aplicando los puntos mencionados anteriormente.

8. BIBLIOGRAFÍA

[1] Comunicados de prensa, Kaspersky. https://www.kaspersky.es/about/press-releases/2019_estan-tus-contrasenas-a-salvo-aumenta-en-un-60-los-usuarios-afectados-por-el-robo-de-contrasenas-en-2019

[2] Página oficial de MATLAB. https://www.mathworks.com/products/matlab.html

[3] Página oficial de Visual Studio Code. https://code.visualstudio.com/

[4] Página oficial de Amazon Web Service. https://aws.amazon.com/es/

[5] Página oficial de Node. https://nodejs.org/es/

[6] Página oficial de NPM. https://www.npmjs.com/

[7] Página oficial de React Native. https://reactnative.dev/

[8] Página oficial de Expo. https://expo.io/

[9] Página oficial de MongoDB. https://www.mongodb.com/

[10] Página oficial de Trello. https://trello.com/

[11] La evolución del 'teletrabajo' en España, en gráficos. https://www.epdata.es/datos/teletrabajo-datos-graficos/517

[12] RFC 4226. https://tools.ietf.org/html/rfc4226

[13] RFC 6238. https://tools.ietf.org/html/rfc6238

[14]Documentación de MongoDB. https://docs.mongodb.com/manual/core/databases-and-collections/

[15] ¿Qué tipos de datos podemos utilizar en MongoDB? https://aula301.com/tipos-datos-podemos-utilizar-mongodb/

[16] Campo auditivo humano. http://www.cochlea.org/es/sonidos/campo-auditivo-humano

[17] Boto3 - The AWS SDK for Python. https://github.com/boto/boto3

[18] Página oficial de ChirpSDK. https://developers-stage.chirp.io/

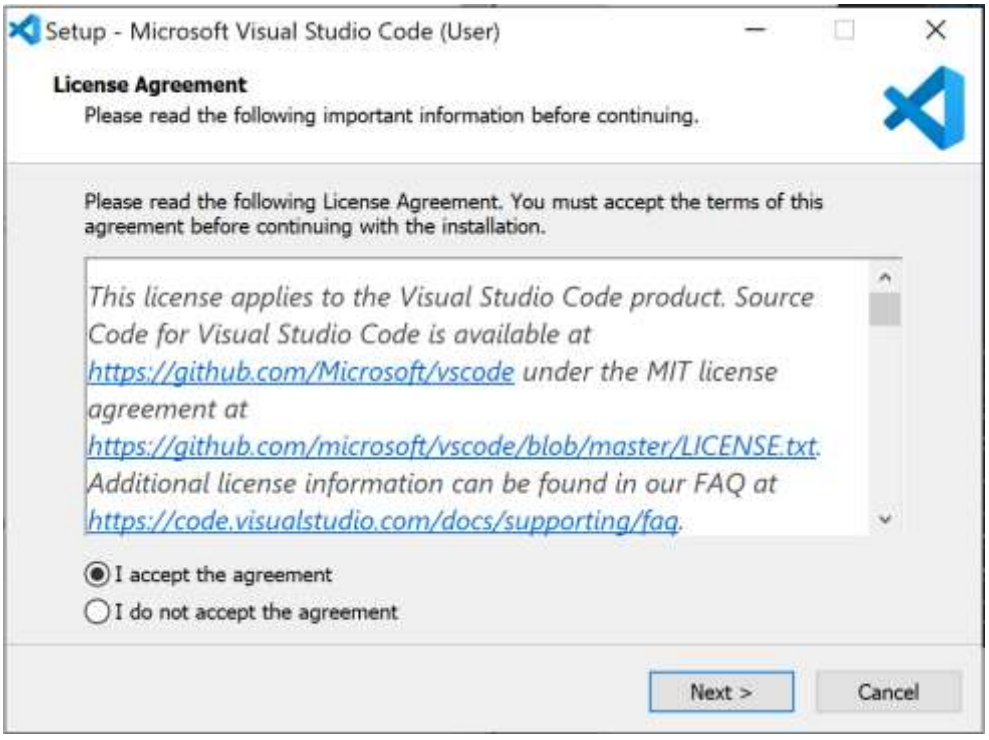
ANEXO A: GUÍA DE INSTALACIÓN

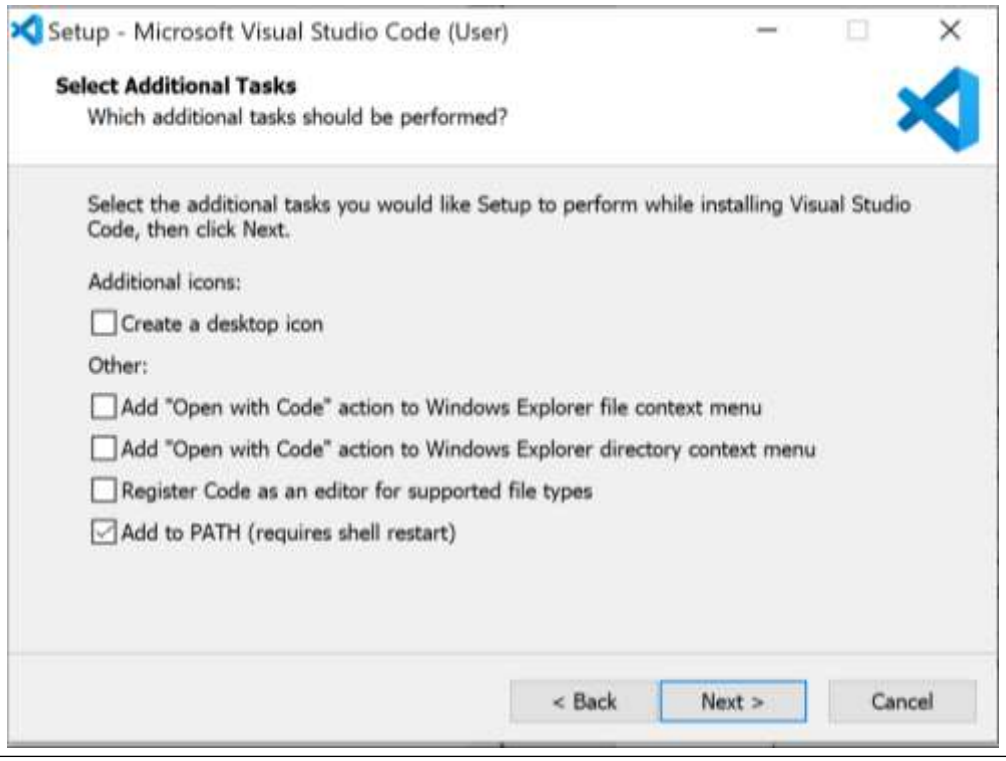
En este Anexo se tratará de realizar una explicación detallada del entorno necesario a instalar para el desarrollo de la aplicación. Para ello, habrá que realizar la instalación de los programas y librerías descritos a continuación.

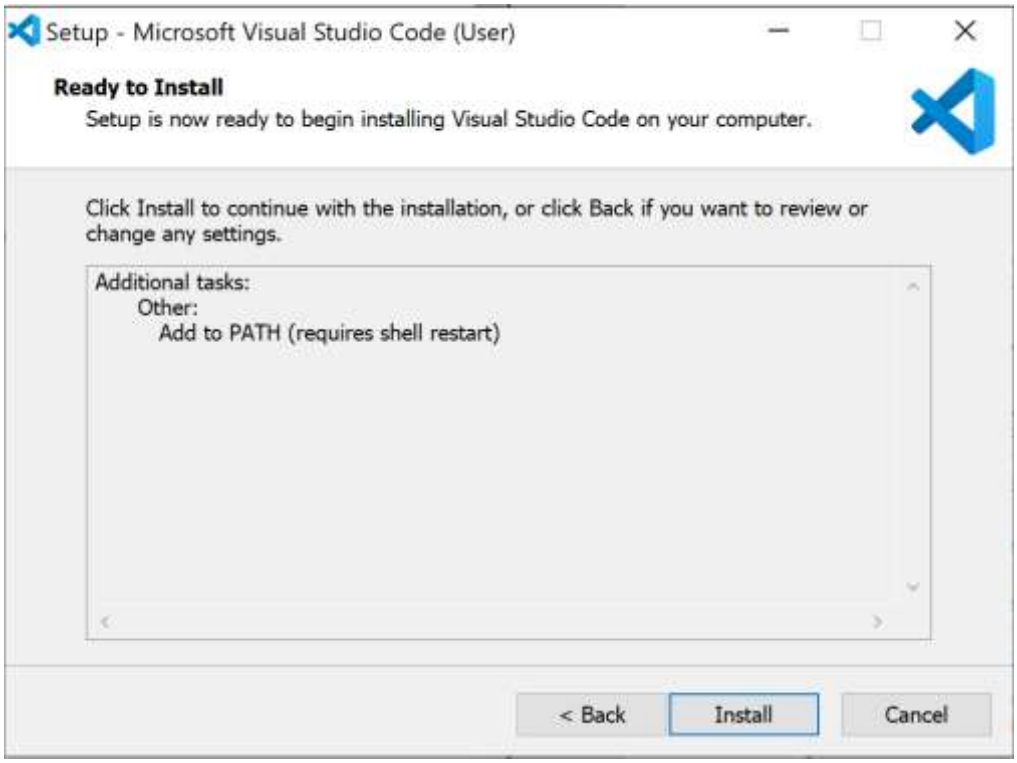
1. Instalación de Visual Studio Code

Visual Studio Code es el entorno de desarrollo integrado necesario para la programación del proyecto. Para instalarlo, se accede a la página oficial de Visual Studio Code [3], y se selecciona la versión para el sistema operativo que estemos utilizando, ya sea macOS, Windows o Linux.

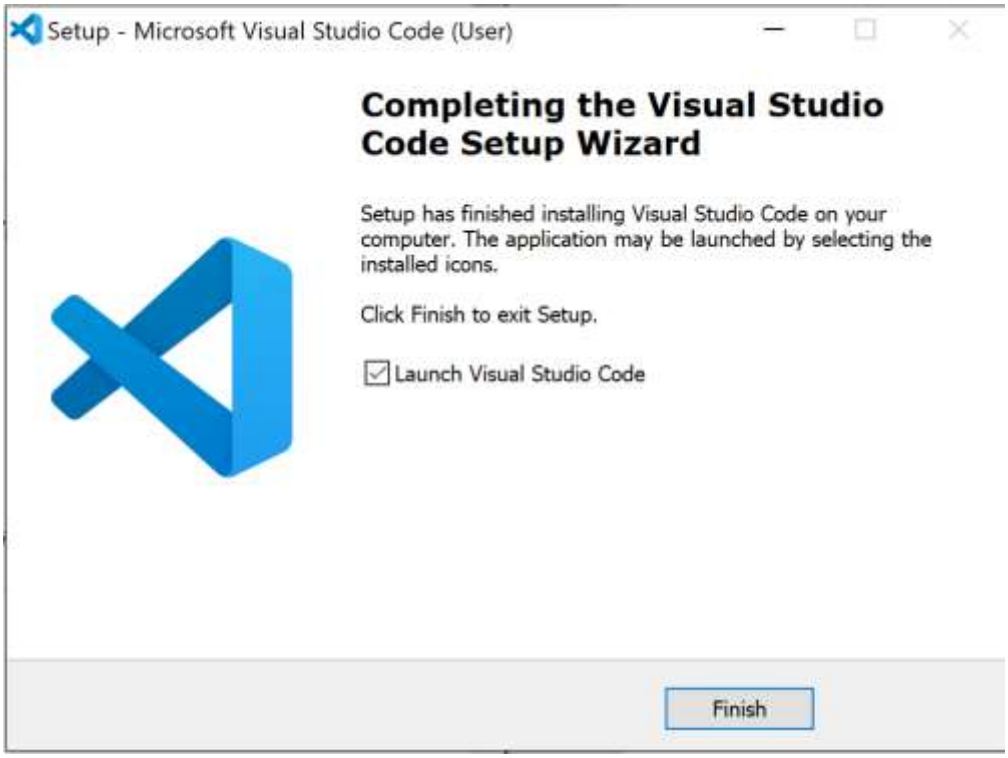
A continuación, se descargará y se abrirá el instalador que mostrará los siguientes pasos a seguir: aceptar el acuerdo de licencia, selección de tareas adicionales, y finalmente se procede a la instalación pulsando “Install”.







Una vez concluye la instalación, se mostrará el siguiente mensaje:



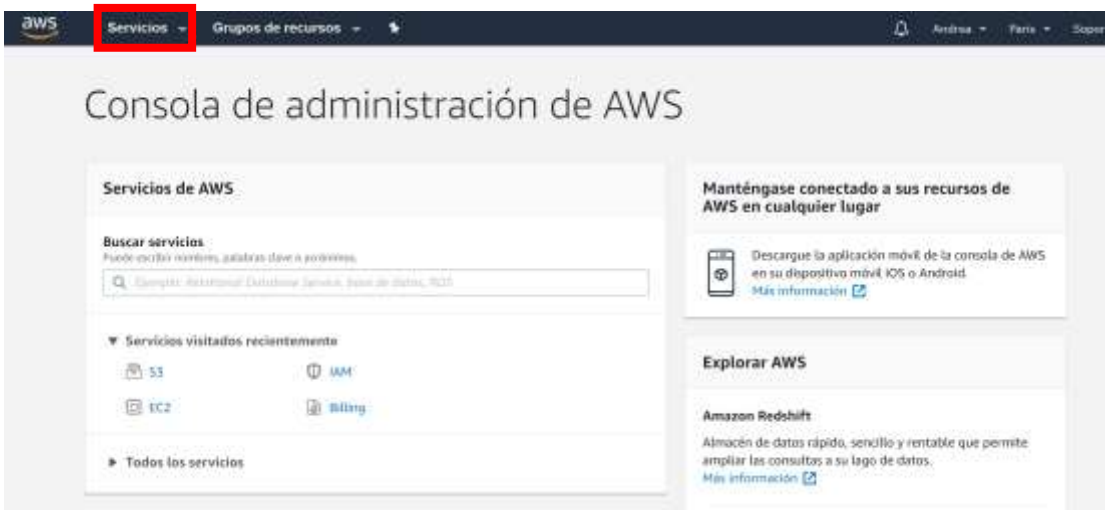
2. Confirguración de Amazon Web Services

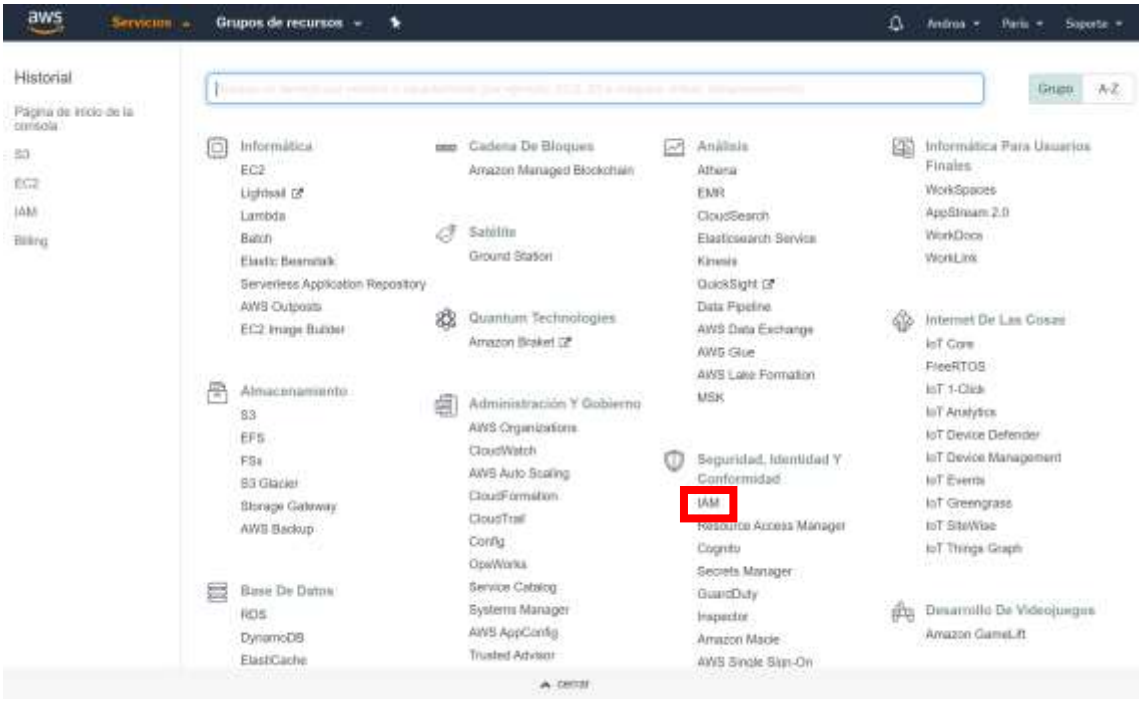
Para configurar el backend en AWS para la aplicación móvil de AudioMark, es necesario prmero descargar los archivos de la carpeta “AWS programs” y la carpeta completa de “python”.

2.1 Para comenzar la puesta a punto en AWS, lo prmero es iniciar sesión en la consola. Para ello, acceda a https://aws.amazon.com y pulse en “Inicie sesión en la consola”.

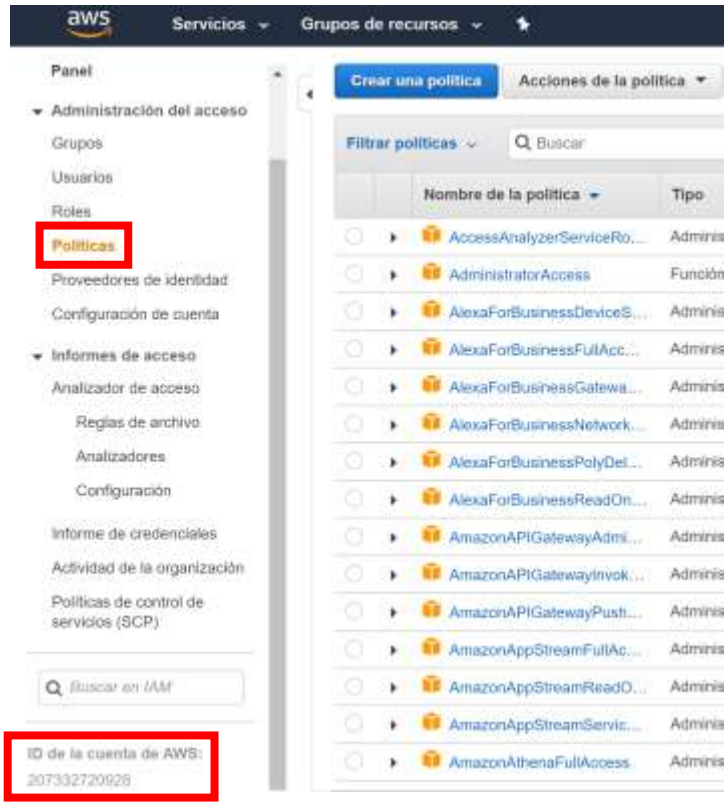
2.2 Si tiene ya una cuenta de usuario raíz en AWS, inicie sesión. Sino, cree una cuenta e inicie sesión.

2.3 Cree un perfil IAM con roles y políticas. Tras iniciar sesión, aparecerá en la consola de administración de AWS. Seleccione el menú de "Servicios". Y seleccione IAM, en la categoría "Seguridad, Identidad y Almacenamiento".





2.4 Seleccione “Políticas” en el menú de la izquierda y anote el ID de su cuenta de AWS.



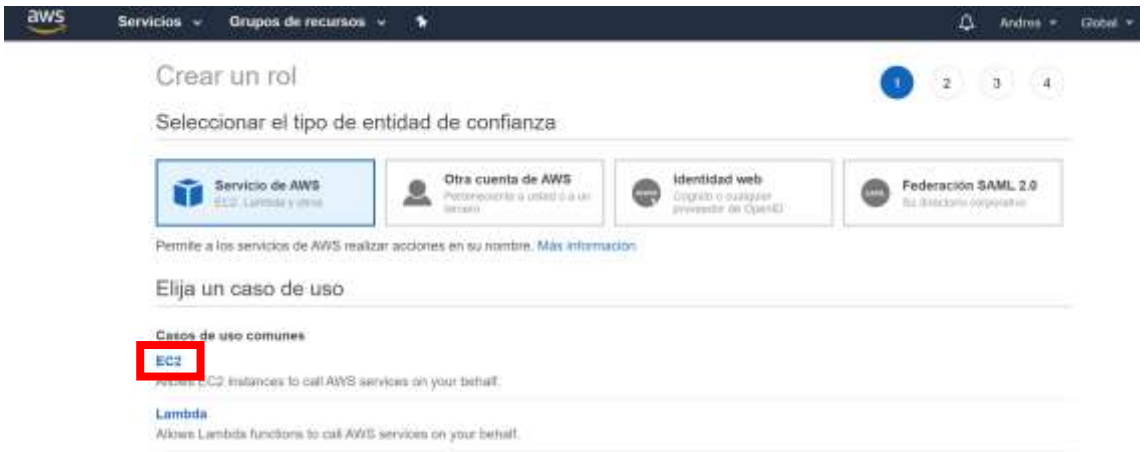
2.5 Seleccione “Crear una política” y la etiqueta de JSON y pegue el siguiente código y guarde la política con el nombre “invokerole”:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "lambda:InvokeFunction",
                "lambda:InvokeAsync"
            ],
            "Resource":"arn:aws:lambda:*:*:function:*"
        }
    ]
}
```

2.6 Repita el paso anterior con el siguiente código y guarde la política con el nombre "lambdapolicy" pero reemplace YOURAWSACCOUNTID con el ID anotado en paso 2.4:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "logs:CreateLogGroup",
            "Resource":          "arn:aws:logs:us-east-
1:YOURAWSACCOUNTID:*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogStream",
                "logs:PutLogEvents"
            ],
            "Resource": [
                "arn:aws:logs:us-east-
1:YOURAWSACCOUNTID:log-group:/aws/lambda/s3test:*"
            ]
        }
    ]
}
```

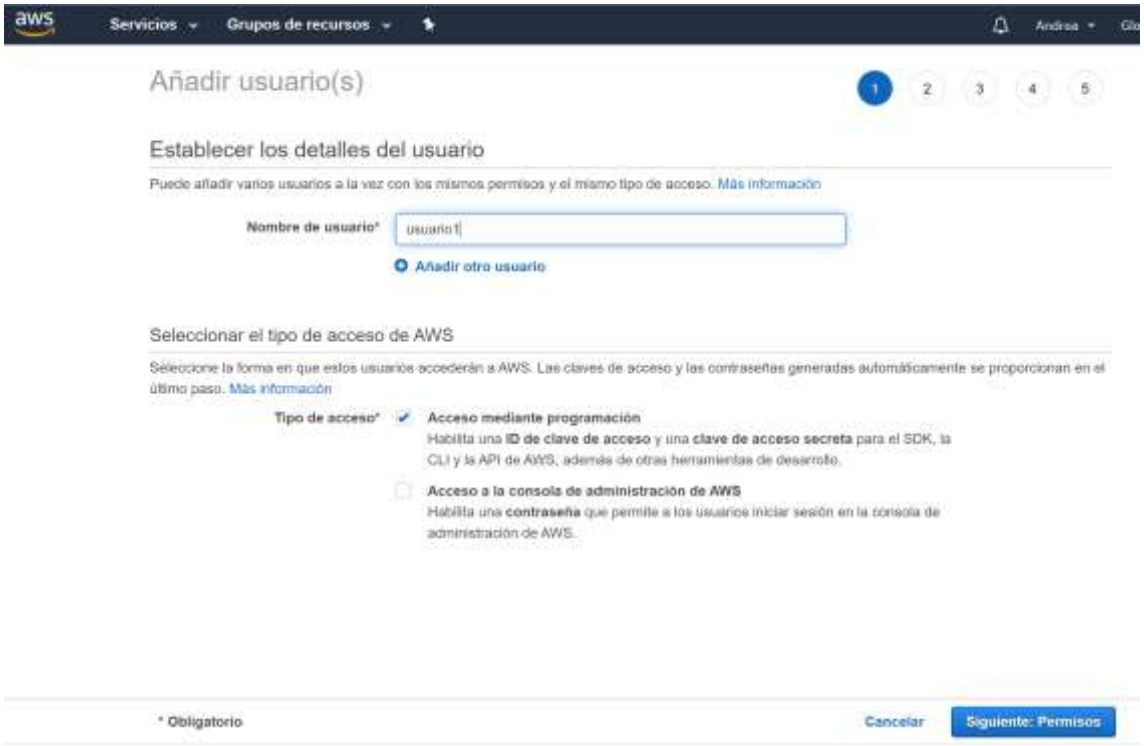
2.7 A continuación, cree un rol al que adjudicar estas políticas en el panel de IAM. Seleccione "crear un rol" y después "EC2".



2.8 Adjudique las siguientes seis políticas: “invokerole”, “lambdapolicy”, “AmazonS3FullAccess”, “AmazonEC2FullAccess”, “AmazonS3ReadOnlyAccess”, “AWSLambdaBasicExecutionRole”.

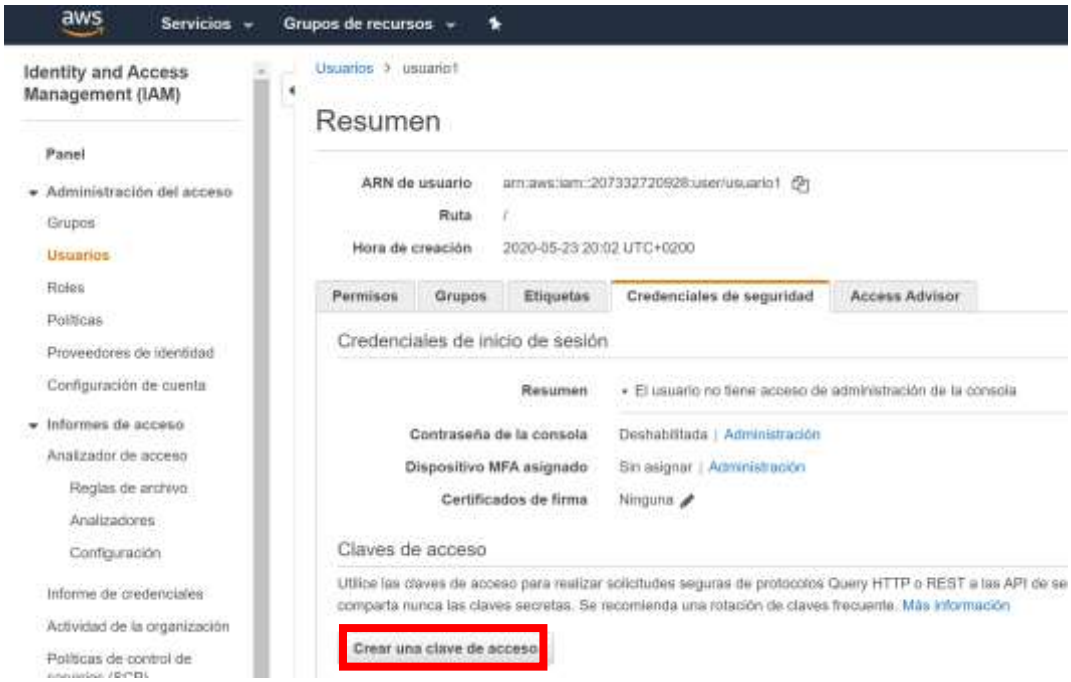
2.9 Continúe hasta que le pidan introducir un nombre para el rol, por ejemplo *AudioMarkRole*. Y haga click en “Crear rol”.

2.10 En el panel de IAM seleccione “Usuarios” y cree usuarios. Añada un nombre y seleccione la primera opción de tipo de acceso (Acceso mediante programación).



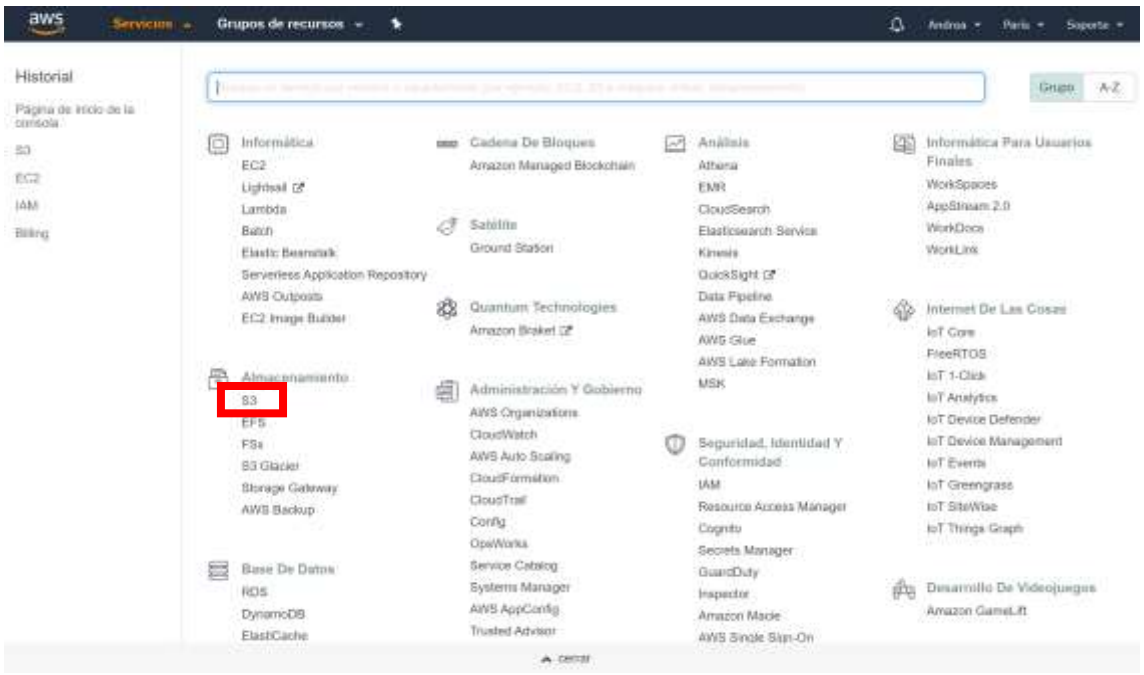
2.11 Haga click en siguiente, adjudique las políticas del paso 2.8 y haga click en siguiente y finalmente en “Crear usuario”.

2.12 Seleccione el usuario creado, vaya a “Credenciales de seguridad” y pulse en “Crear una clave de acceso” y a continuación descargue el archivo .csv.


aws
Servicios
Grupos de recursos
Identity and Access Management (IAM)
Panel
Administración del acceso
Grupos
Usuarios
Roles
Políticas
Proveedores de identidad
Configuración de cuenta
Informes de acceso
Analizador de acceso
Reglas de archivo
Analizadores
Configuración
Informe de credenciales
Actividad de la organización
Políticas de control de
Usuarios > usuario1
Resumen
ARN de usuario
arn:aws:iam::207332720928:user/usuario1
Ruta
/
Hora de creación
2020-05-23 20:02 UTC+0200
Permisos
Grupos
Etiquetas
Credenciales de seguridad
Access Advisor
Credenciales de inicio de sesión
Resumen
El usuario no tiene acceso de administración de la consola
Contraseña de la consola
Deshabilitada | Administración
Dispositivo MFA asignado
Sin asignar | Administración
Certificados de firma
Ninguna
Claves de acceso
Utilice las claves de acceso para realizar solicitudes seguras de protocolos Query HTTP o REST a las API de se
comparta nunca las claves secretas. Se recomienda una rotación de claves frecuente. Más información
Crear una clave de acceso

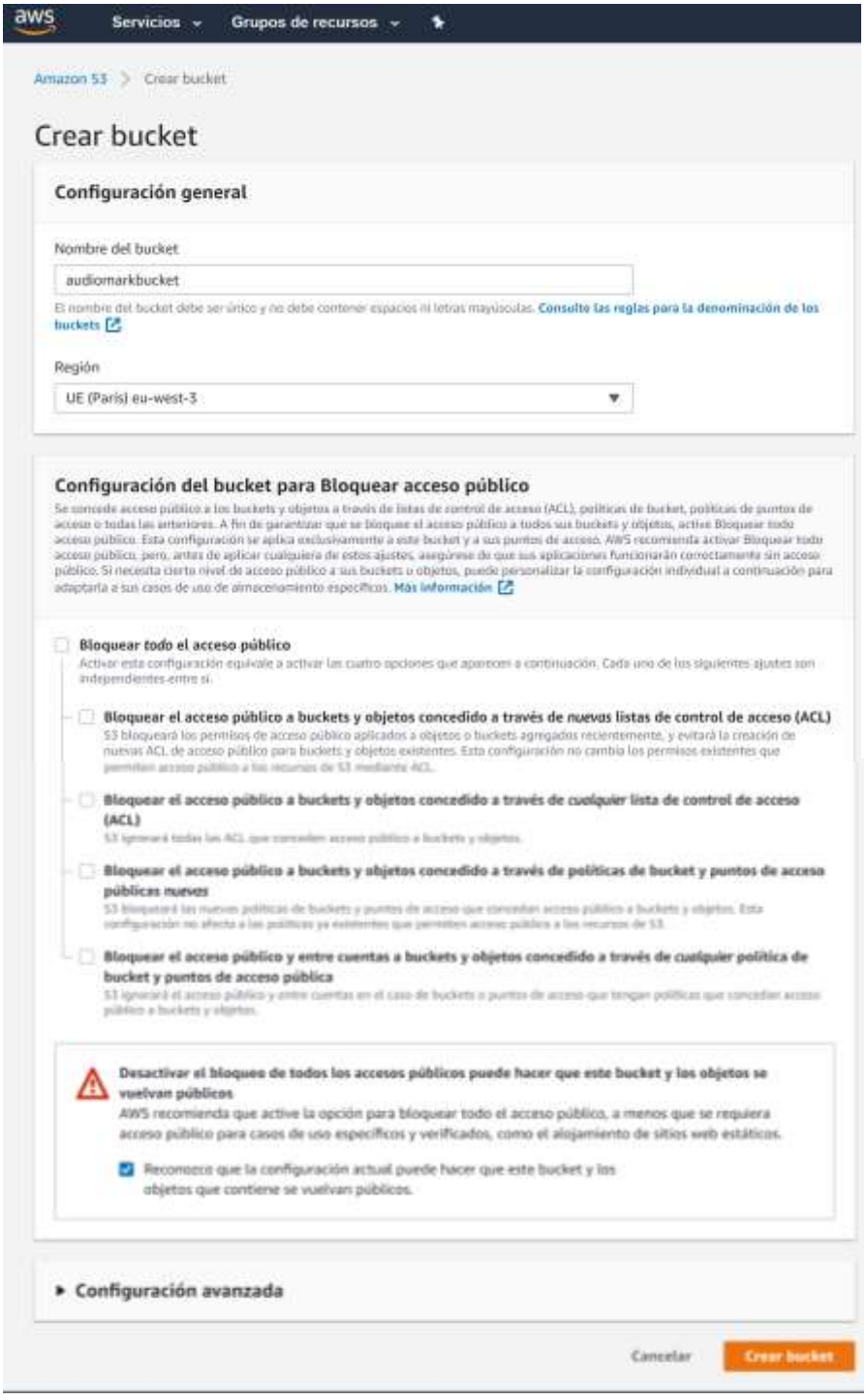

Crear una clave de acceso
Correcto
Esta es la solo vez que se pueden ver o descargar las claves de acceso secretas. No puede recuperarlas más adelante. Sin embargo, puede crear nuevas claves de acceso en cualquier momento.
Descargar archivo .csv
ID de clave de acceso
Clave de acceso secreta
AKIATARP6IEQFZAMITXI
******** Mostrar
Cerrar

2. 13 Seleccione el servicio S3 en la categoría Almacenamiento.



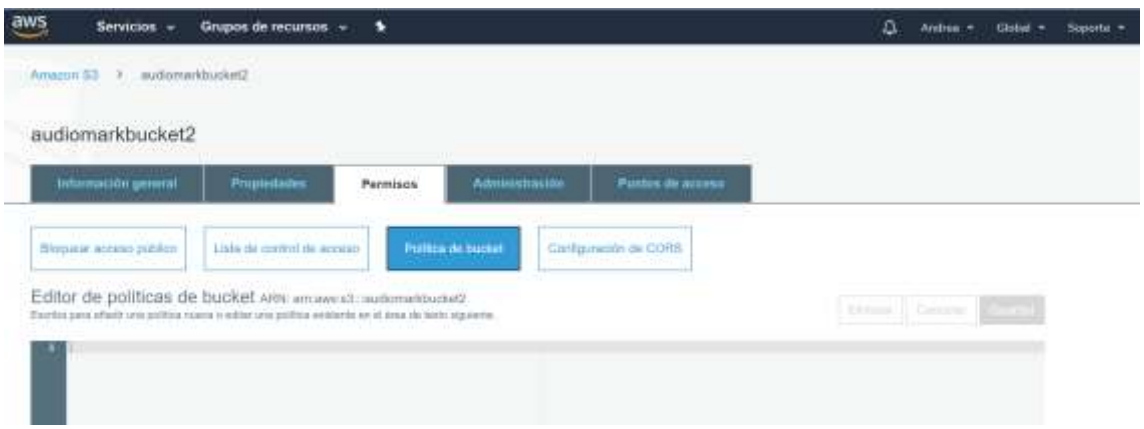
2.14 Seleccione “Crear bucket”

2.15 Introduzca un nombre para el bucket, *audiomarkbucket*, por ejemplo. Seleccione su región y permita todo acceso público. Finalmente, pulse en “Crear bucket”.



2.16 Tras haber creado el bucket, seleccionelo, vaya a “Permisos” y seleccione “Política de bucket” y pegue lo siguiente en el editor de políticas de bucket:

```
{
     "Version": "2008-10-17",
     "Statement": [
          {
               "Sid": "AllowPublicRead",
               "Effect": "Allow",
               "Principal": {
                    "AWS": "*"
               },
               "Action": [
                    "s3:GetObject",
                    "s3:Get*",
                    "s3:List*"
               ],
               "Resource": "arn:aws:s3:::audiomarkbucket/*"
          }
     ]
}
```



2.17 Guarde e ignore en caso de que aparezcan warnings.

2.18 A continuación, instale la interfaz de línea de comandos AWS cli2.

- Utilice este link si su sistema operativo es Windows: https://docs.aws.amazon.com/cli/latest/userguide/install-cliv2-windows.html
- Utilice este link si su sistema operativo es Linux: https://docs.aws.amazon.com/cli/latest/userguide/install-cliv2-linux.html
- Utilice este link si su sistema operativo es macOS: https://docs.aws.amazon.com/cli/latest/userguide/install-cliv2-mac.html

2.19 Configure AWS cli2: https://docs.aws.amazon.com/cli/latest/userguide/cli-chap-configure.html, para ello necesitará la clave de acceso obtenida en el paso 2.12 en el archivo .csv y la región del bucket introducida en el paso 2.15.

2.20 Copie el archivo *chirprcpath.py* de la carpeta "AWS programs" y ejecutelo desde el símbolo del sistema (cmd).

2.21 Copie el archivo *chirprc* de la carpeta "python" en la ubicación obtenida tras realizar el paso anterior.

2.22 Navegue hasta el directorio /Documents/python\ChirpSDK-Python-3.5.0 (1) 2 y ejecute `python ultra_embed.py`. Seguramente aparecerán errores por falta de módulos. Para solucionar esto, instale→

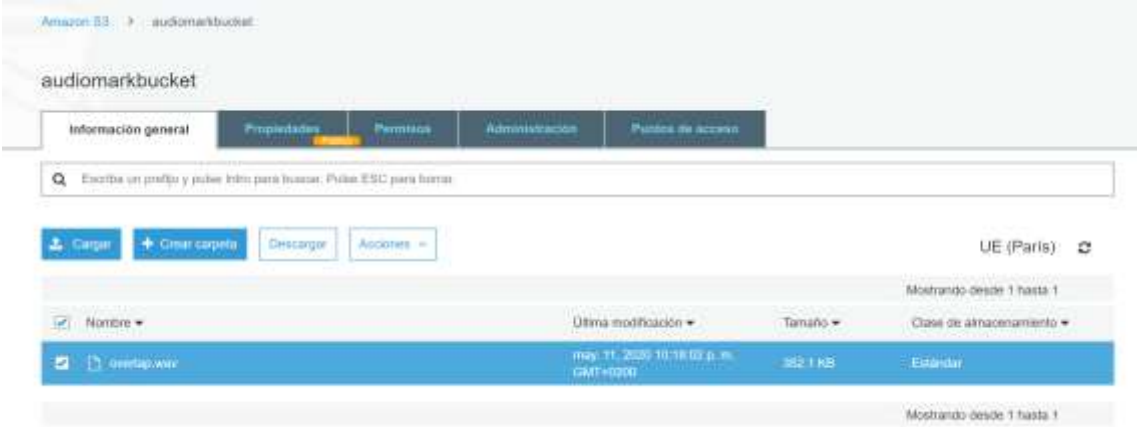
- "Scipy" con el comando: `pip3 install scipy`
- "Matplotlib" con el comando: `pip3 install matplotlib`
- "Librosa" con el comando: `pip3 install librosa`

Una vez hecho esto, vuelva a ejecutar `python ultra_embed.py` y compruebe que el archivo *overlap.wav* en esa misma ubicación, ha cambiado su fecha y hora de última modificación.

2.23 Instale "boto3"→ `pip3 install boto3`

2.24 Ejecute `python upload.py`

2.25 Acceda al bucket S3 y compruebe que el archivo está ahí. Deberá ver algo como esto:



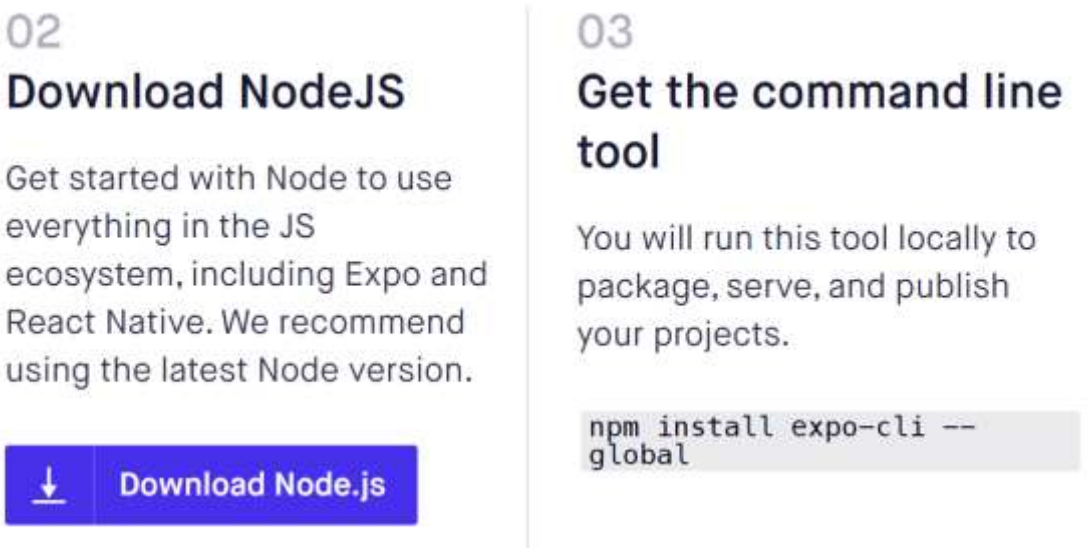
2.26 Copie la URL del objeto y peguela en la URL de la línea 40 de /my-new-project/screens/audioscreen.js.

3. Configuración de la aplicación

Para poner a punto la aplicación móvil siga los siguientes pasos:

3.1 Descargue el directorio *my-new-project* en su PC.

3.2 Acceda a https://expo.io/learn en el PC y siga los pasos 2 y 3:

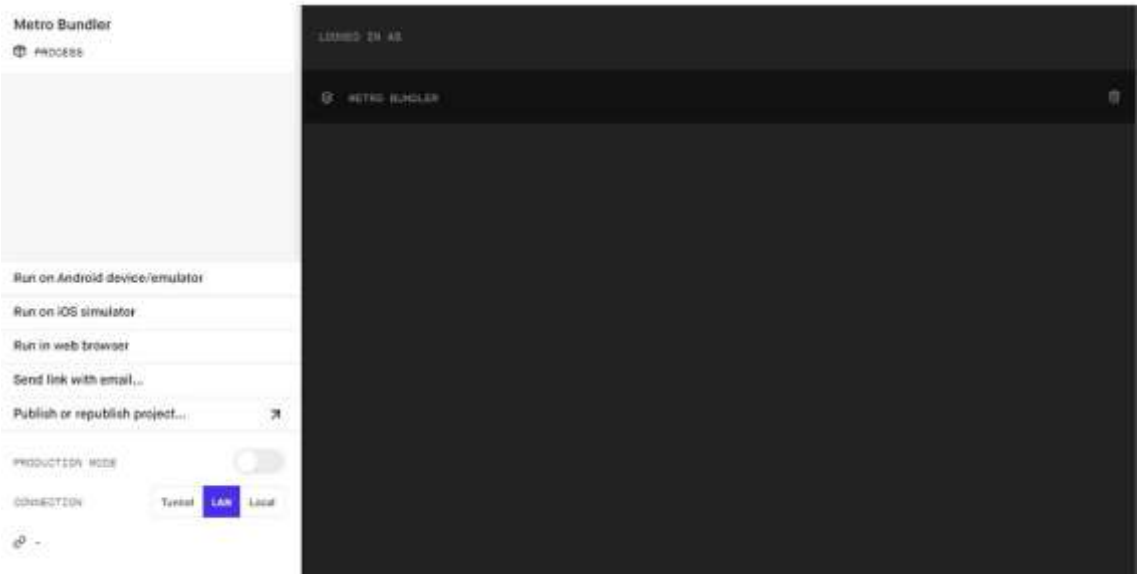


No hará falta instalar NPM por separado, ya que se distibuye con Node.js. Cuando descargue Node.js, NPM se instalará automáticamente en su PC.

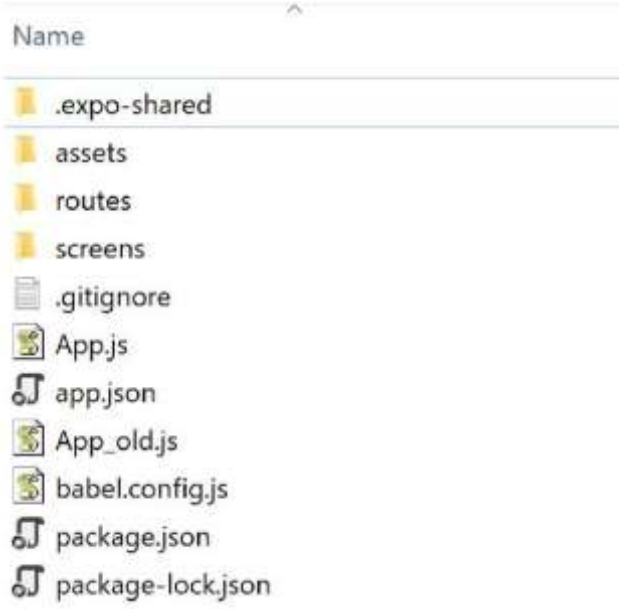
3.3 En línea de comandos, ejecute lo siguiente:

```
expo init my-new-project
cd my-new-project
expo start
```

3.4 Esto abrirá en el naegador web la siguiente ventana:



3.5 Cierre el navegador y pare el proceso en línea de comandos (Ctrl+C). Copie la carpeta *my-new-project* que descargó en el paso 3.1. Su nueva carpeta *my-new-project* debería tener un aspecto similar a este:



3.6 Cree una cuenta en Expo o inicie sesión en una ya existente ejecutando `expo login`. Guarde esta información de inicio sesión para más adelante.

3.7 Instale todos los módulos de NPM mediante el siguiente comando en línea de comandos: `npm install all`.

3.8 Relance de nuevo la aplicación: `expo start`.

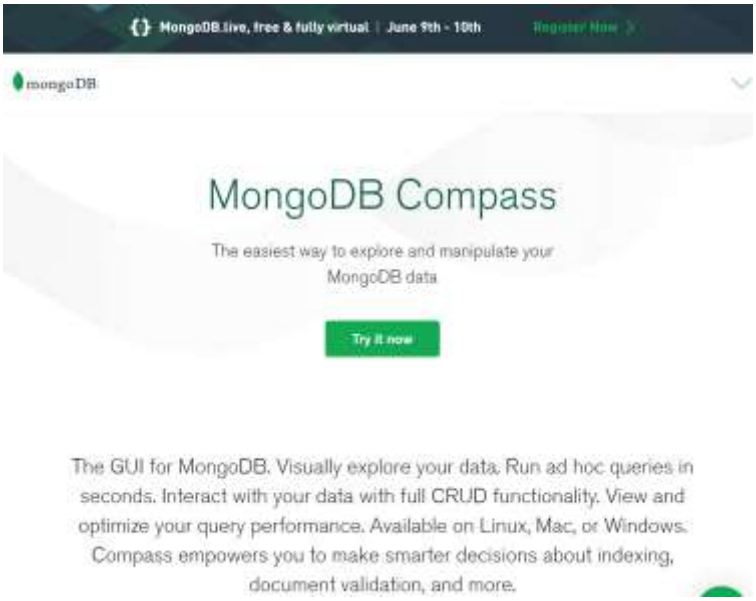
3.9 Descargue en su móvil la aplicación "Expo Client" de su tienda de aplicaciones correspondiente.

3.10 Acceda a la cuenta de expo creada en el paso 3.6 y deberá aparecer en "Projects→ Recently in development" el proyecto.

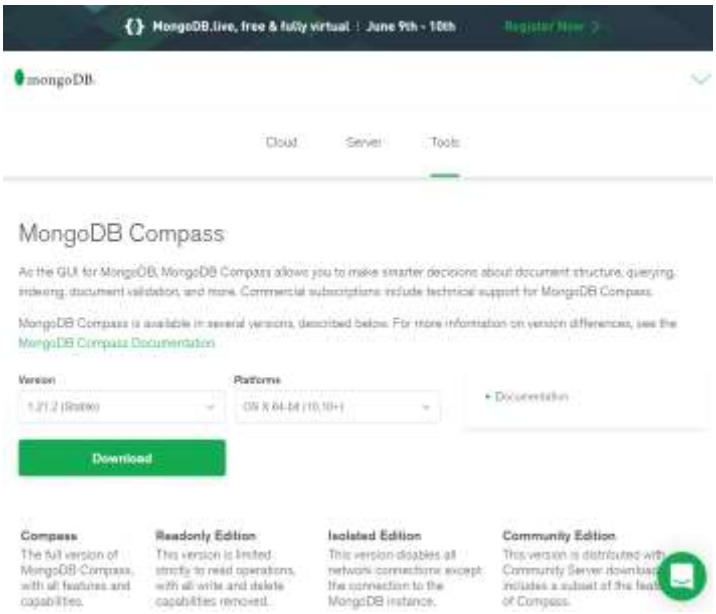
3.11 Seleccione el proyecto y tras cargar, debería aparecer en la pantalla de inicio de sesión de la aplicación de AudioMark.

4. Instalación de MongoDB Compass para acceder a las BBDD de MongoDB

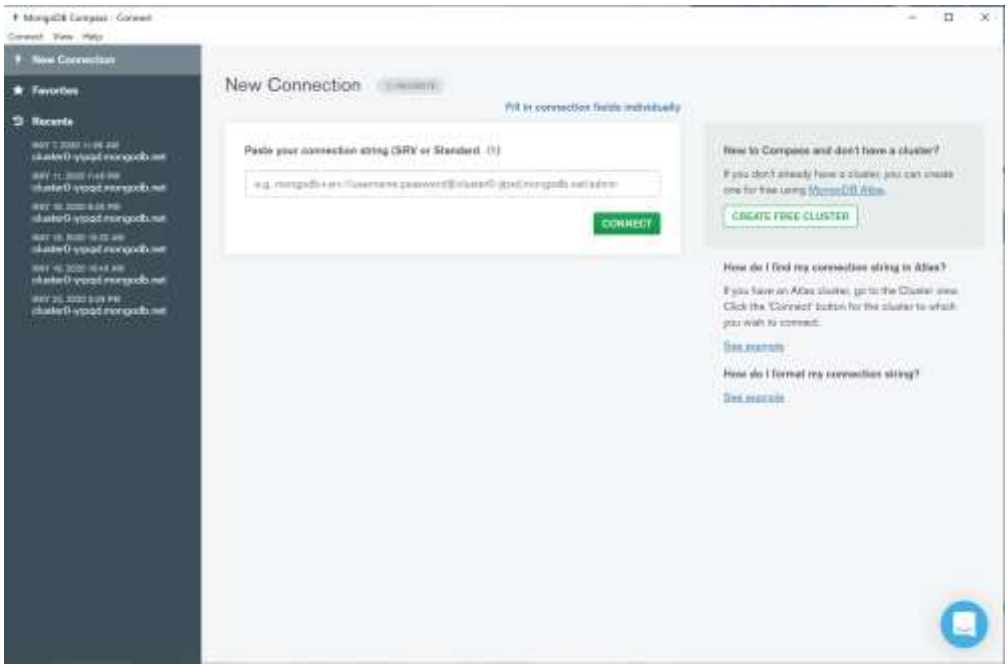
4.1 Acceda a https://www.mongodb.com/products/compass y pulse en "Try it now".



4.2 Seleccione la última versión estable en "Version" y su sistema operativo en "Platforms". Y haga click en "Download".



4.3 Tras un par de minutos, se descarga y se instalará automáticamente y aparecerá la siguiente pantalla que indica que MongoDB Compass se ha instalado correctamente:



ANEXO B: MANUAL DE USUARIO

En este Anexo se procede a explicar detalladamente cómo se pone en marcha el proyecto AudioMark y qué uso debe el usuario hacer de este, paso a paso.

1. Puesta en marcha de la aplicación móvil

Para poner en funcionamiento la aplicación móvil, primero hay que escribir el siguiente código en el terminal:

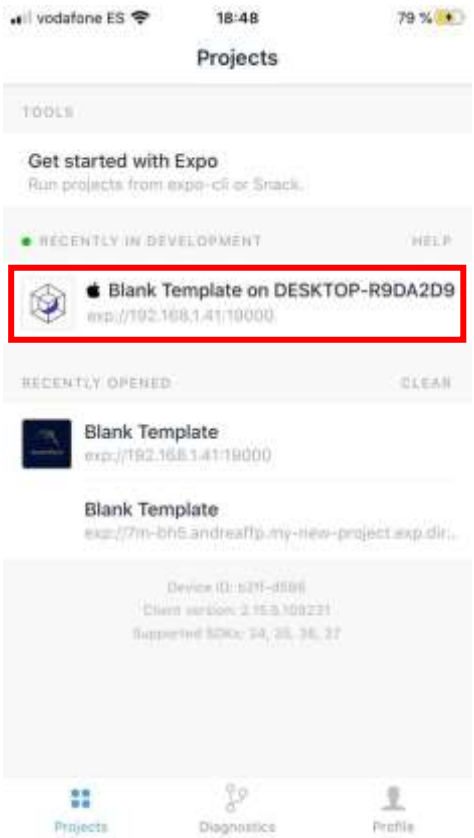
```
C:\Users\Andrea>cd C:\Users\Andrea\Desktop\BU\EC 463 Senior Design I\20-26-Watermark-master ENTREGA FINAL\python\ChirpSD
K-Python-3.5.0 (1) 2\my-new-projectPRUEBA
```

Y a continuación:

```
C:\Users\Andrea\Desktop\BU\EC 463 Senior Design I\20-26-Watermark-master ENTREGA FINAL\python\ChirpSDK-Python-3.5.0 (1)
2\my-new-projectPRUEBA>expo start
```

En el terminal deberá aparecer algo como esto:

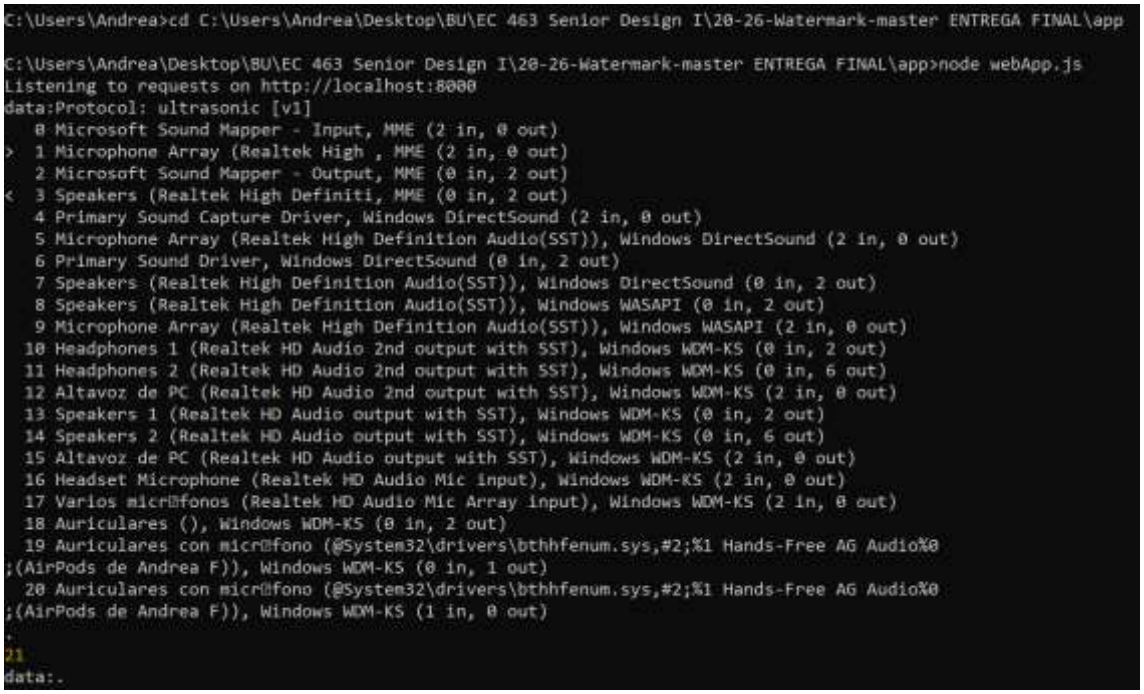
Ahora la aplicación móvil de AudioMark está lista para abrirla desde la aplicación Expo Client en nuestro teléfono móvil. Para ello acceda a la aplicación Expo Client y seleccione el proyecto que aparece en "Recently in development".



Por último, inicie sesión en la aplicación de AudioMark introduciendo "User1" en el campo de usuario y "Test1" como contraseña. La aplicación estaría lista para reproducir el audio encriptado.

2. Puesta en marcha del servidor

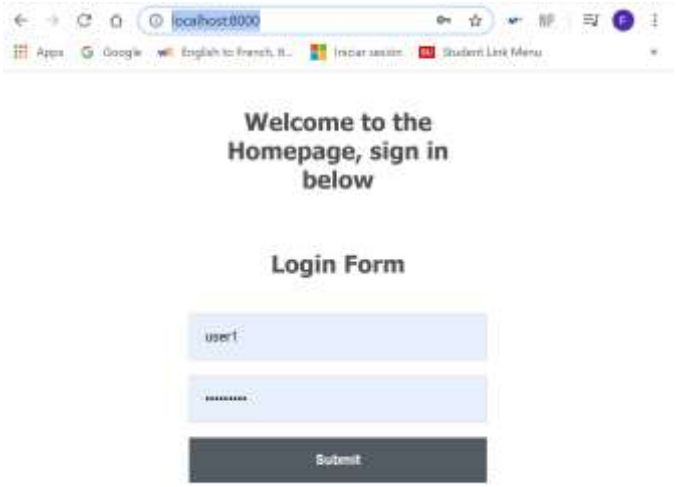
Para arrancar el servidor, que lleva a cabo varias funciones entre las que destaca la puesta en marcha de la página web y la modificación del código por TOTP, debe seguir los siguientes pasos:



Con esto, se arranca el servidor, que comienza a escuchar en busca del ultrasonido con el código. Además, la página web está lista para introducir los campos de nombre de usuario y contraseña en el puerto 8000.

3. Acceda a la página web

En el navegador, acceda a http://localhost:8000 y se mostrará esta página:



4. Cómo realizar el proceso de autenticación

Introduzca el siguiente nombre de usuario: “user1” y contraseña: “password1”. Pero aún no pulse “Submit”.

Antes deberá reproducir el tono encriptado para que el servidor lea el segundo factor de autenticación.

Para ello siga los siguientes comandos:

- Vaya al directorio *...\python\ChirpSDK-Python-3.5.0 (1) 2*

```
C:\Users\Andrea>cd C:\Users\Andrea\Desktop\BU\EC 463 Senior Design I\20-26-Watermark-master ENTREGA FINAL\python\ChirpSD
K-Python-3.5.0 (1) 2
```

- Ejecute *python ultra_embed.py*

```
C:\Users\Andrea\Desktop\BU\EC 463 Senior Design I\20-26-Watermark-master ENTREGA FINAL\python\ChirpSDK-Python-3.5.0 (1) 2>
python ultra_embed.py
```

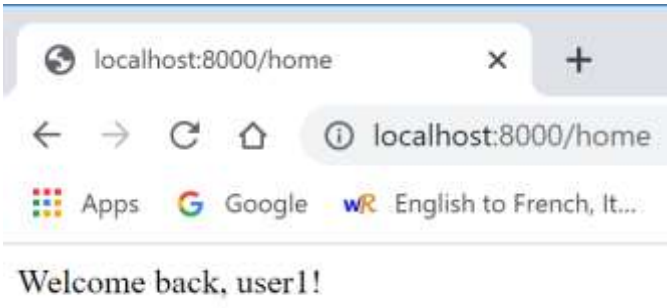
- Ejecute *python upload.py*

```
C:\Users\Andrea\Desktop\BU\EC 463 Senior Design I\20-26-Watermark-master ENTREGA FINAL\python\ChirpSDK-Python-3.5.0 (1) 2>
python upload.py
```

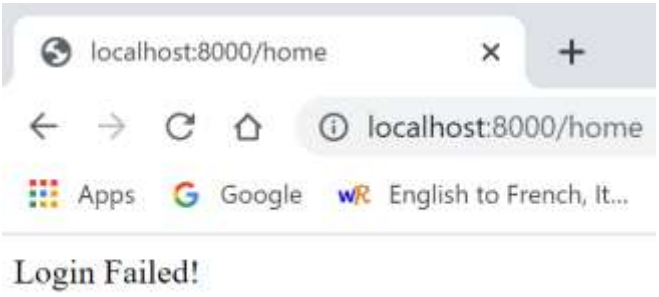
Una vez haya ejecutado estos comandos, el audio estará listo para ser reproducido desde el botón “Play/Pause” de la aplicación móvil de AudioMark frente a la página web de autenticación.

Una vez haya reproducido el audio completo, pulse el botón de “Submit”.

Si el proceso ha sido satisfactorio, se mostrará la siguiente pantalla:



En caso contrario la pantalla mostrará el siguiente mensaje:



ANEXO C: ALINEACIÓN DEL PROYECTO CON LOS OBJETIVOS DE DESARROLLO SOSTENIBLE

Los Objetivos de Desarrollo Sostenible (ODSs) son un conjunto de 17 objetivos diseñados para ser un "plano para lograr un futuro mejor y más sostenible para todos".

AudioMark mantiene gran relación con algunos de esos objetivos como se muestra en la siguiente tabla:

Dimensión ODS	ODS identificado	Rol
Económica	ODS 8: Crecimiento económico y trabajo decente	Secundario
Industrial	ODS 9: Industria, innovación e infraestructuras	Secundario
Medio ambiental	ODS 13: Acción climática	Primario
Social	ODS 16: Paz, Justicia e Instituciones sólidas	Primario

En la dimensión económica, AudioMark favorece el crecimiento económico ya que evita grandes fugas de información al proporcionar una capa de seguridad adicional en la autenticación. De esta manera ahorra millones de euros que sería necesario invertir para remediar posibles daños causados por el robo de información sensible en las compañías. Con este proyecto, esa información está guardada de forma más segura y como consecuencia favorece al desempeño laboral en un ambiente de mayor confianza como se define en el octavo Objetivo de Desarrollo Sostenible.

Por otro lado, este proyecto se alinea con el Objetivo de Desarrollo Sostenible 9, ya que la industria se desarrollará gracias a la disponibilidad de un entorno más seguro. Creando un método alternativo al envío del segundo factor de autenticación vía SMS, que es el más usado hoy en día, AudioMark aporta innovación a un ámbito como el de la

ciberseguridad que está hoy en día en auge. En otras palabras, explorar otras formas de autenticación de dos factores, como se ha hecho con este proyecto, provocará que otros métodos sean investigados para facilitar a los usuarios su uso, llamando así a la innovación.

Uno de los ODS que más se relaciona con el trabajo es el número 13, relacionado con el cambio climático. Con el cambio radical que ha sufrido el ámbito laboral por el coronavirus, digitalizándose y pasando de ser presencial a teletrabajo en un alto porcentaje, AudioMark favorecerá que muchos de los empleados que no se pueden permitir realizar el trabajo de forma remota, debido a la confidencialidad con la que deben tratar los datos, puedan hacerlo, pues el proceso de autenticación será más seguro. Así, muchas más personas podrán evitar el trabajo presencial cuando sea necesario, lo que disminuirá los desplazamientos de casa a la oficina y, por consiguiente, la contaminación. Desde que comenzó la cuarentena en España los niveles de NO_2, un gas altamente contaminante generado sobre todo por la combustión de combustibles fósiles y los escapes de los vehículos a motor, han bajado día tras día. En la siguiente gráfica se muestra como cuando empezó la cuarentena disminuyó la contaminación por debajo del límite establecido por la Organización Mundial de la Salud.

Por último, en el ámbito social, AudioMark se relaciona con el Objetivo de Desarrollo Sostenible 16 que trata la paz, justicia y lucha por unas instituciones sólidas. Esto se verá favorecido por la robusta seguridad de la información de las compañías o los individuos y abogando por un uso responsable de esta, haciendo que sólo las personas indicadas puedan accederla.