Universitat Oberta de Catalunya

uoc.edu

# Covid Tracing

**Autor: Jordi Querol Marco.**
Máster Universitario en Seguridad de las Tecnologías de la información y las comunicaciones.
Área: Protocolos y aplicaciones de seguridad.

**Director: Richard Rivera Guevara.**
**Responsable de área: Víctor García Font.**

29/12/2020

**Agradecimientos:**

A mi esposa y familia, por su soporte y paciencia.

A mis profesores, y alumnos, por sus palabras de ánimo y soporte.

# Resumen

El presente trabajo de fin de máster (TFM), se desarrolla con el objetivo de crear una aplicación para el sistema operativo Android, que ayude a diagnosticar la Covid-19 de una forma precoz a toda aquella persona que la instale en su dispositivo móvil, notificarle si en algún momento determinado, en función de la distancia y tiempo que se haya encontrado con otras personas, haya podido estar expuesta o no a la enfermedad.

La metodología para realizar esta tarea se fundamenta en respetar la privacidad de las personas, mediante la comunicación entre dispositivos móviles sin conexión previa, sin transferir información sensible, y proporcionando la confianza para que, en caso sean notificadas por posible riesgo de contagio, puedan decidir por sí mismas realizarse un diagnóstico médico fehaciente, que determine si han contraído la enfermedad o no.

Tras realizar diversas pruebas con los resultados obtenidos, se ha podido comprobar que la aplicación cumple sus funciones correctamente. Por ello, no implica que esta pueda estar exenta de errores, pero si es un buen punto de partida para seguir mejorando e innovando.

**Palabras clave:** Covid Tracing, Covid-19**,** SARS-CoV-2.

# Abstract

This final master thesis (FMT) is developed with the aim of creating an application for the Android operating system, which helps to diagnose Covid-19 in early way to anyone who installs it on their mobile device, notify you if at any given time, depending on the distance and time that you have met with other people, you may or don't have been exposed to the disease.

The methodology to carry out this task is based on respecting the privacy of people, through communication between mobile devices without a previous connection, without transferring sensitive information, and providing the confidence so that, in case they are notified of a possible contagion risk, they can decide for themselves to make a reliable medical diagnosis, which determines whether they have contracted the disease or do not.

After carrying out various tests with the results obtained, it has been possible to verify that the application fulfils its functions correctly. Therefore, it does not imply that it can be free of errors, but it is a good starting point to continue improving and innovating.

**Keywords:** Covid Tracing, Covid-19**,** SARS-CoV-2.

# Índice

# 1. Introducción

Según información oficial de la **OMS [1] (Organización Mundial de la Salud)**,el 31 de diciembre de 2019 se detectó un conglomerado de casos de neumonía en la ciudad de Wuhan (provincia de Hubei, China), que posteriormente se determinó dichos casos eran causados por un nuevo coronavirus, conocido actualmente como Covid-19. A fecha actual 16 de septiembre de 2020, y comienzo de este trabajo, el virus se encuentra extendido a nivel mundial ocasionando una pandemia de aproximadamente 30,6 millones de personas infectadas, y más de 953 mil fallecidas, a pesar de todos los esfuerzos conjuntos de las autoridades y ciudadanía para frenarlo, las cifras siguen creciendo día a día.

A continuación, se mencionará porque se realiza el presente trabajo, objetivos que pretenden conseguirse, la existencia de proyectos que tratan de ayudar a resolver problemas relativos a la pandemia, las posibilidades existentes referente a herramientas para el enfoque y metodología de trabajo, metodología adoptada, planificación de trabajo y herramientas a utilizar para el desarrollo del proyecto, donde se expondrá:

- Hardware mínimo necesario.
- Software para el funcionamiento del proyecto.
- Protocolos de seguridad.

Acto seguido, se expondrán los requisitos funcionales de la aplicación, los resultados obtenidos del proyecto, y finalmente las conclusiones.

## 1.1. Contexto y justificación del Trabajo

Actualmente, atravesamos una situación a nivel social compleja a causa de una pandemia provocada por la Covid-19. Se advierte un descontrol de contagios severo, y en consecuencia, es necesario establecer mecanismos que proporcionen certidumbre a la población, para que pueda entenderse cuando una persona debe realizarse un diagnóstico de la enfermedad Covid-19, de forma veraz, principalmente para prevenirla, o detectarla precozmente y paliar los síntomas, ya que aún no existe una vacuna efectiva.

Aunque actualmente existen aplicaciones desarrolladas para dispositivos móviles de rastreo de contactos entre la población, dichas aplicaciones, se encuentran sujetas a políticas de protección de privacidad distintas según la legislación de los diversos países donde se aplican, además de otras restricciones como, por ejemplo, funcionales en base al sistema operativo bajo el cual operan, la transparencia insuficiente por la que se rige el funcionamiento de la misma, las escasas descargas por parte de los usuarios por la

desconfianza a que datos sensibles puedan ser revelados, exceso en el consumo de energía de la batería de los dispositivos y etc.

Por dichos motivos y con la intención de mejorar las aplicaciones existentes. Este trabajo se basa en el desarrollo de una aplicación para dispositivos móviles Android, la cual ayudará a todas aquellas personas que hagan uso de ella, para colaborar voluntariamente a hacerse un diagnóstico de la enfermedad, en caso de que esta detecte que ha podido producirse riesgo de contagio, y así se lo notificará al usuario.

## 1.2. Objetivos del Trabajo

Los principales objetivos de este trabajo son los siguientes:

**Objetivos de investigación:**

- Determinar a ciencia cierta el tiempo de exposición y distancia suficientes, para que una persona pueda contagiar la Covid-19 a otra que no la sufre.
- Determinar el tiempo necesario que una persona infectada de Covid-19, debe permanecer en aislamiento.

**Objetivo de creación:**

- Crear una aplicación para dispositivos móviles Android.

**Objetivos de la aplicación:**

- Mejorar donde las aplicaciones existentes están fallando.
- Llegar al máximo de usuarios posibles que dispongan dispositivos móviles con sistema Android.
- Verificar de forma fehaciente y privada si el usuario se encuentra afectado por Covid-19 o no.
- Sugerir al usuario una prueba de diagnóstico si ha podido estar en contacto estrecho con alguna persona infectada, sea asintomática o no.
- Ayudar a diagnosticar la enfermedad Covid-19 de forma precoz.
- Privacidad, seguridad y anonimato del usuario.

# 2. Estado del arte

Actualmente, existen diversas aplicaciones ya realizadas por diversas entidades para abordar el problema planteado, por ejemplo:

- **Aplicación de rastreo en China [2] (publicada oficialmente):** siendo una aplicación de instalación obligatoria, proporcionando a los usuarios códigos QR con diversos estados, basándose en su historial de viajes para comprobar y calificarles según sus contactos, especialmente si han contactado con casos positivos. Todo lo mencionado implica monitorizar las actividades de los usuarios.

- **Aplicación de rastreo Suiza [3] (Swiss Covid, publicada oficialmente):** siendo una aplicación de instalación voluntaria. Basándose en un rastreo de proximidad para determinar quién ha estado en proximidad física con una persona Covid-19 positivo, sin revelar la identidad del contacto o donde ocurrió. Transmite continuamente una identificación pseudoaleatoria que representa el teléfono del usuario, y registra las identificaciones pseudoaleatorias observadas de los teléfonos cercanos. Cuando a un paciente se le diagnostica Covid-19, puede cargar los id pseudoaleatorios enviados previamente desde su teléfono, a un servidor central. Las aplicaciones de otros usuarios pueden usar los datos del servidor para estimar localmente si han estado expuestos al virus por proximidad física. Si la aplicación detecta un alto riesgo, notifica al usuario.

- **Aplicación Española Radar Covid [7 (publicada oficialmente):** Al abrir la aplicación, realiza una “llamada” a un servidor solicitando un **token** (ver anexo) de usuario. Estos **tokens** permiten identificar al terminal (no a la persona) y son autogenerados por el servidor. Cuando se obtiene el **token**, lo utiliza tanto para pedirle al servidor los datos de los textos que muestra la app en la pantalla de bienvenida, como para solicitar los datos de configuración de **DP3T** (ver anexo) al servidor. Requiere al usuario activar la configuración necesaria (**Bluetooth** y **localización**), y tras sincronizar datos, envía notificaciones a los usuarios sobre posibles contactos positivos, siempre que se hayan reportado previamente los casos positivos a la autoridad correspondiente mediante un código de 12 dígitos. Además, desde una perspectiva técnica, determina un riesgo de exposición basándose en los datos obtenidos del servidor.

- **Aplicación Alemana Corona Warn [9] (publicada oficialmente):** la aplicación utiliza el Bluetooth para medir si los usuarios de teléfonos móviles han estado cerca durante un largo período de tiempo, y almacena esta información en el teléfono. Lo hace posible con la ayuda de claves criptográficas, llamadas ID, que los teléfonos se

envían y reciben constantemente. Las identificaciones se borran después de dos semanas. Si alguien da positivo en un test de coronavirus e introduce ese resultado en la aplicación, ésta se encarga de enviar un mensaje a todos los que hayan estado cerca de la persona infectada. Todo sucede de forma anónima y voluntaria.

# 3. Enfoque y método seguido

Para llevar a cabo los objetivos que se pretende alcanzar con la aplicación a desarrollar, existen tecnologías implícitas para este proyecto, y posibilidades diversas que se explican a continuación.

## 3.1. Aplicación Cliente

A continuación, se exponen requisitos implícitos de la aplicación cliente, opciones disponibles de comunicación para Bluetooth, y conexión de cliente a servidor.

### 3.1.1. Permisos Bluetooth Low Energy (Requisito implícito)

Para usar las características Bluetooth en la aplicación, deben habilitarse los permisos para el uso del dispositivo Bluetooth para establecer cualquier comunicación.

### 3.1.2. Comunicación por Bluetooth de bajo consumo (Bluetooth Low Energy)

A partir del sistema Android 4.3 (API nivel 18) incorpora la compatibilidad de la plataforma con **Bluetooth de bajo consumo [10]** (BLE) en función central, y ofrece una API que puede utilizarse para descubrir dispositivos, consultar por servicios y transmitir información.

Los casos de uso típicos son:

- Transferir volúmenes de datos reducidos entre dispositivos cercanos.

- Interactuar con sensores de proximidad como las balizas, que permiten que el usuario tenga una experiencia personalizada en función de su ubicación actual.

A diferencia del Bluetooth clásico, Bluetooth de bajo consumo está diseñado para ofrecer un consumo de energía significativamente

menor. Esto permite que las apps de Android se comuniquen con dispositivos BLE con requisitos de consumo más estrictos.

Las siguientes son las funciones y responsabilidades que se aplican cuando un dispositivo Android interactúa con un dispositivo BLE:

- **Central y periférica:** El dispositivo con la función central escanea en busca de paquetes de información, y el dispositivo con la función periférica transmite los paquetes de información.

- **Servidor GATT y cliente GATT:** determina cómo se comunican entre sí los dispositivos una vez establecida la conexión.

## 3.2. Modelos de intercambio de información

En los siguientes subapartados se detalla las posibilidades existentes para realizar la transmisión de datos.

### 3.2.1. Modelo de transmisión (Broadcast model)

El **modelo de transmisión [10]**, aprovecha el mecanismo de envío de paquetes que contienen información concreta para el rastreo de contactos, difundiendo estos periódicamente para que otros dispositivos recopilen dicha información entrante.
Cada paquete de información transmitido contiene 31 bytes de información, que de estos, 3 bytes se reservan para información de estructura, dejando 28 bytes para datos de información de seguimiento de contactos. Cada estructura de envío contiene un tipo de encabezado de longitud de 2 bytes, lo que causa la reducción del espacio disponible. Si solo se utiliza una estructura de datos, puede transmitirse una total de 26 bytes de datos dedicados al rastreo de contactos.

Se puede utilizar varios tipos de estructura de datos para transportar los datos de rastro de contactos:

- Una opción se basa en utilizar un servicio asociado dedicado. Esta requiere anunciar el servicio, además de los datos del servicio, proporcionando 20 bytes para los datos de rastreo de contactos.

- Otra opción consta en utilizar una estructura de datos específica del fabricante diseñada para transportar datos personalizados. Por tanto, se utiliza un solo tipo de estructura, limitando la sobrecarga del encabezado a 2 bytes, la estructura del fabricante

debe comenzar con el ID de la empresa de 2 bytes, dejando 25 bytes para los datos de seguimiento de contactos.

### 3.2.2. Modelo de conexión

El **modelo de conexión [10]**, se basa en la conexión entre 2 dispositivos para realizar el intercambio de información de rastreo de contactos. Cada dispositivo anuncia su existencia al incluir el **UUID (Universally Unique Identifier)** del servicio correspondiente a los paquetes enviados. Tras la detección de un dispositivo cercano que reconoce el servicio, se establecerá una conexión para intercambiar datos de seguimiento de contactos con el dispositivo remoto.

La mayoría de las soluciones de este modelo de conexión, intercambian datos utilizando el perfil ATT (Attribute Protocol), donde cada dispositivo expone su servicio, y los datos de seguimiento de contactos se transfieren escribiendo y leyendo atributos de este servicio en el dispositivo remoto. La cantidad de datos que pueden intercambiarse en este modelo de conexión, solo queda limitado por la duración de la conexión.

Los términos clave para este modelo son:

- **Perfil de atributos genéricos (GATT):** es una especificación para enviar pequeños fragmentos de datos, denominados atributos, a través de un vínculo BLE.

- **Protocolo de atributos (ATT):** GATT está basado en el protocolo de atributos (ATT), por eso también se denomina GATT/ATT. Está optimizado para funcionar en dispositivos BLE. Por esta razón, usa la menor cantidad de bytes posible. Cada atributo se identifica de forma exclusiva mediante un identificador único universal (UUID), que es un formato estándar de 128 bits empleado para identificar información de manera exclusiva. Los atributos transportados por ATT tienen formato de características y servicios.

- **Característica:** Una característica contiene un solo valor, y 0 a n descriptores que describen el valor de una característica. Puede interpretarse como un tipo, similar a una clase.

- **Descriptor:** Los descriptores son atributos definidos que describen el valor de una característica.

- **Servicio:** Un servicio es una colección de características.

### 3.2.3. Modelo híbrido

El **modelo híbrido [10]**, se basa en el modelo de transmisión y el modelo de conexión usados de forma simultánea. En este modelo, los dispositivos que no admiten el modelo de transmisión, utilizarán el modelo de conexión para intercambiar los datos de seguimiento de contactos, y otros modelos utilizarán el modelo de transmisión.

## 3.3. Conexión cliente y servidor

### 3.3.1. OKHTTP

**OkHttp [14]** es un cliente HTTP eficiente de forma predeterminada:

- Es compatible con Android 5.0+ (API level 21+) y Java 8+.
- El soporte HTTP / 2 permite que todas las solicitudes al mismo host compartan un socket.
- La agrupación de conexiones reduce la latencia de las solicitudes.
- El almacenamiento en cache de respuesta evita las solicitudes repetidas.
- Persevera cuando la red es problemática:
    - Se recupera de forma silenciosa de problemas de conexión comunes.
    - Si el servicio tiene varias direcciones IP, intentará direcciones alternativas si falla la primera conexión.
    - Admite funciones TLS modernas como TLS 1.3, ALPN, fijación de certificados (certificate pinning).

## 3.4. Funciones requeridas en segundo plano

- **La aplicación debe permanecer activa**, ya sea en primer o segundo plano, la aplicación debe funcionar constantemente en todas aquellas situaciones en las que el usuario se encuentre en cercanía con otras personas ajenas o no a su entorno.

- **Debe transmitir paquetes de datos de forma periódica**, para que otros dispositivos móviles que se encuentren cercanos puedan detectar datos de rastreo de contactos, la aplicación debe transmitir paquetes de información de forma periódica.

- **Debe transmitir paquetes de datos a un servidor y recibir notificaciones**, para conectarse a un servidor, es necesario realizarlo de forma que permita transmitir información cifrada, para evitar que esta pueda quedar comprometida por terceros.

- **Recibir paquetes de datos de forma periódica**, para que otros dispositivos móviles que se encuentren cercanos puedan recibir datos de rastreo de contactos, la aplicación debe ser capaz de detectar constantemente los paquetes de información relativos al servicio establecido de la aplicación.

- **Privacidad de la información**, los paquetes de datos que se intercambien en todo momento, no deben revelar información sensible relativa al usuario de la aplicación, ni permitir que el dispositivo pueda ser rastreable.

## 3.5. Frameworks posibles para el Servidor (Back-End)

En los siguientes apartados se exponen posibles frameworks aplicables como solución para la lógica del servidor.

### 3.5.1. Express (Node.js/JavaScript)

**Express [16],** es un framework flexible y minimalista para **Node.js [16],** el cual, es un entorno en tiempo de ejecución multiplataforma, de código abierto, para la capa de servidor basado en el lenguaje de programación JavaScript, asíncrono, con entrada / salida de datos en una arquitectura orientada a eventos y basado en el motor Versión 8 de

Google. Fue creado con el enfoque de ser útil en la creación de programas de red altamente escalables, como por ejemplo, servidores web. Al contrario que la mayoría del código JavaScript, no se ejecuta en un navegador, sino en el servidor. Funciona con un modelo de evaluación de un hilo de ejecución, usando entradas y salidas asíncronas, las cuales pueden ejecutarse concurrentemente en un número de hasta cientos de miles sin incurrir en costos asociados al cambio de contexto. Este diseño de un único hilo de ejecución entre todas las solicitudes atiende a necesidades de aplicaciones altamente concurrentes, donde toda operación que realice entradas y salidas, debe tener una función de devolución de llamada (Callback).

**Express**, proporciona un conjunto de características robusto para aplicaciones web y móviles, y entrega valiosos métodos de utilidades HTTP y middleware.
Node.js puede ser combinado con una base de datos documental o no relacional (MongoDB o CouchDB) y JSON, lo que permite desarrollar en un entorno de desarrollo JavaScript unificado.
Express es popular por facilitar la migración de programadores web de JavaScript del lado cliente al desarrollo del lado servidor, es eficiente con los recursos, es minimalista, ya que los componentes que se deseen utilizar deben instalarse específicamente para su uso.

### 3.5.2. Django

**Django [16]**, es un framework de desarrollo web de código abierto, escrito en Python, que respeta el patrón de diseño conocido como MVC (Modelo Vista Controlador). Incluye un servidor liviano para realizar pruebas y trabajar en la etapa de desarrollo. En un entorno de desarrollo, no se necesita un web server instalado, ya que trae su propio servidor liviano para este propósito, con la restricción de permitir un solo usuario a la vez, y opera con las bases de datos PostgreSQL, MySQL, Oracle o SQLite. Permite a los desarrolladores la construcción de Web APIs navegables, compatibles con lectura de datos ORM (Object Relational Mapping) o no ORM, y con políticas de autenticación basadas en paquetes como OAuth1 y OAuth2.

### 3.5.3. Ruby on Rails (Ruby)

**Ruby on Rails [16]**, es un framework web escrito para el lenguaje de programación Ruby. Como Django, proporciona mecanismos estándar para el enrutado de URLs, acceso a bases de datos, generación de plantillas y formateo de datos como JSON o XML. Promueve el uso de patrones de diseño como DRY (Don’t Repeat Yourself), MVC (Modelo Vista Controlador) y otros.

### 3.5.4. ASP.NET

**ASP.NET [16],** es un framework web de código abierto desarrollado por Microsoft para construir aplicaciones y servicios modernos. Se pueden crear rápidamente sitios web basados en HTML, CSS, y JavaScript, escalarlos para usarse por millones de usuarios y añadir capacidades complejas como APIs web, formularios de datos o comunicaciones en tiempo real. Permite a los programadores escribir código ASP.NET usando cualquier lenguaje .NET soportado.

### 3.5.5. Mojolicious (Perl)

**Mojolicious [16],** es un framework web de nueva generación para el lenguajede programación Perl. Ofrece Framework web en tiempo real, para crecer desde prototipos de un solo fichero hasta aplicaciones web MVC estructuradas, rutas RESTful, plugins, comandos, plantillas específicas de Perl, negociación de contenidos, gestión de sesiones, validación de formatos, framework de pruebas, servidor de ficheros estáticos, soporte Unicode, implementación cliente / servidor equipada de HTTP y WebSocket con IPv6, TLS, SNI, IDNA, HTTP/SOCKS5 proxy, UNIX domain socket, Comet (long polling), keep-alive, connection pooling, timeout, cookie, y soporte de compresión multipart y gzip, convertidores JSON y HTML/XML y generadores con soporte de selector CSS, portable y API orientada a objetos y Perl. Código basado en años de experiencia, gratis y de código abierto.

## 3.6. Opciones para bases de datos

Será necesario almacenar datos de diagnóstico específicos, mensajes aleatorios con marca de tiempo de una forma segura y eficiente. Las opciones existentes para cumplir dichos requisitos se comentan a continuación.

### 3.6.1. MySQL

**MySQL [17]**, es un sistema de gestión de bases de datos relacional, desarrollado bajo licencia dual: Licencia pública general / Licencia comercial por Oracle Corporation.

Existen varias interfaces de programación de aplicaciones diversas, que permiten acceder a las bases de datos MySQL, incluyendo C, C++, C#, Java, Perl, PHP, Python, Ruby y muchos más.
Se utiliza mucho en aplicaciones web, su popularidad como aplicación web está muy ligada a PHP. Es una base de datos rápida en la lectura cuando utiliza el motor no transaccional, pero puede provocar problemas de integridad en entornos de alta concurrencia en la modificación. Cualquier entorno que sea el que se vaya a utilizar MySQL, es importante monitorizar de antemano el rendimiento para detectar y corregir errores, tanto de SQL como de programación.

## 3.6.2. Microsoft SQL Server

**Microsoft SQL Server [18]**, es un sistema de gestión de bases de datos relacional, desarrollado por Microsoft. Algunas de sus características son:

- Soporta transacciones.
- Procedimientos almacenados.
- Incluye entorno gráfico de administración.
- Permite trabajar en modo cliente / servidor.
- Permite administrar información de otros servidores de datos.

Para conectarse a SQL Server se necesita un login usuario a nivel de servidor. Cuando la política de seguridad se define como Windows authentication y el servidor se combina con las definiciones del dominio, los logins se definen en el **active directory**.
A nivel de base de datos el usuario se identifica como un User qué está relacionado generalmente al login, y los privilegios User existen solamente en el ámbito de la base de datos. Para otorgar derechos generales puede asistirse con listas de Server roles o database roles, cada cual, con privilegios específicos a un rol específico, y cada usuario asociado con uno de estos roles obtienen los privilegios asociados con él.
Algunas herramientas posibilitan crear réplicas parciales o completas de las bases de datos, mejorar la disponibilidad, y recuperar de desastres.

## 3.6.3. PostgreSQL

**PostgreSQL [19]**, es un sistema de gestión de bases de datos relacional orientado a objetos y de código abierto, publicado bajo la licencia postre SQL. No tiene un gestor de defectos, haciendo muy difícil conocer el estado de sus defectos.

Algunas de sus características son:

- Alta concurrencia, permite que mientras un proceso escribe en una tabla, otros accedan a la misma tabla sin necesidad de bloqueos.
- Permite claves ajenas (foreign Keys).
- Procedimientos almacenados.
- Vistas.
- Integridad transaccional.
- Herencia de tablas.
- Tipos de datos y operaciones geométricas.
- Soporte para transacciones distribuidas.
- Permite seguridad en términos generales.
- Integridad en restricciones en el dominio.
- Integridad referencial.
- Disparadores.
- Autorizaciones.
- Conexión al sistema de gestión de bases de datos.
- Transacciones y respaldos.

Los bloques de código que se ejecutan en el servidor pueden ser escritos en varios lenguajes, desde las operaciones básicas de programación, como bifurcaciones y bucles, hasta las complejidades de la programación orientada a objetos con la programación funcional.

### 3.6.4. MongoDB

**MongoDB [20]**, es un sistema de bases de datos NoSQL, orientado a documentos y de código abierto. Guarda estructuras de datos BSON (una especificación similar a JSON) con un esquema dinámico, haciendo que la integración de los datos en ciertas aplicaciones sea más fácil y rápida.
Algunas características de MongoDB son:

- Soporta búsqueda por campos cómo consultas de rangos y expresiones regulares.

- Cualquier campo en un documento puede ser indexado, también es posible hacer índices secundarios.

- Soporta replicación.

- Puede ejecutarse en múltiples servidores, balanceando la carga y/o replicando los datos para poder mantener el sistema funcionando en caso de que exista un fallo de hardware. Pueden agregarse nuevos servidores con el sistema de base de datos funcionando.

- Puede ser utilizado como un sistema de archivos.

- Proporciona una función MapReduce, qué puede ser utilizada para el procesamiento por lotes de datos y operaciones de agregación.

- Permite realizar consultas utilizando JavaScript, haciendo que éstas sean enviadas directamente a la base de datos para ser ejecutadas.

- Soporta una diversidad de drivers oficiales para diversos lenguajes de programación.

MongoDB es adecuada para, almacenamiento y registros de eventos, sistemas con alto volumen de lecturas, aplicaciones móviles, almacén de datos operacional de sitios web, proyectos que utilizan metodologías de desarrollo iterativo o ágiles, y otros casos diversos.

## 3.7. Metodología adoptada

Mencionadas las opciones diversas anteriores, se opta por realizar una aplicación nueva, utilizando de forma implícita la tecnología **Bluetooth de bajo consumo** para el sistema Android, puesto que facilita el ahorro de energía en los dispositivos móviles, y el modelo a utilizar para la comunicación y transmisión de datos entre los dispositivos, se basa en el **modelo de transmisión**, donde la función de los dispositivos será **central y periférica de forma simultánea**.

Para la conexión al servidor, se utilizará el cliente **OkHttp,** puesto que facilita la conexión segura entre cliente y servidor pudiendo utilizar **certificado digital**, y por tanto, el uso del protocolo **TLS** (Transport Secure Layer) de conexión segura.

Por otro lado, la parte del servidor, se utilizará **Express (Node.js/JavaScript)**, dado que es un framework que permite instalar solamente los módulos específicos a utilizar, proporciona robustez para aplicaciones web y móviles, y aporta soluciones útiles para HTTP y HTTPS, además de la configuración de middlewares para mayor seguridad.
La base de datos a utilizar será **MongoDB**. Como se menciona anteriormente, nos interesa por aspectos necesarios para esta aplicación, como almacenamiento y registros de eventos, sistemas con alto volumen de lecturas y el uso para aplicaciones móviles.

La metodología a seguir consta de una aplicación que realizará un protocolo descentralizado mediante el modelo de transmisión, y otro centralizado que actuará conjuntamente con el protocolo descentralizado. A continuación, se detallan las funciones a realizar:

### 3.7.1. Protocolo centralizado

- **Pregunta al usuario**, verifica si ha sido diagnosticado por una entidad sanitaria de forma fehaciente del virus Covid-19, comprobando contra un servidor la veracidad del diagnóstico.

- **Cada cierto tiempo determinado**, la aplicación enviará los mensajes pseudoaleatorios anónimos transmitidos a otros usuarios, a un servidor. A su vez, transmitirá los mensajes pseudoaleatorios y anónimos recibidos para la verificación por parte del servidor mediante mensaje y marca de tiempo. Si el usuario ha podido estar en contacto con alguna persona infectada de Covid-19 en un rango de tiempo especificado, el servidor le devolverá una notificación conforme ha estado expuesto.

### 3.7.2. Protocolo descentralizado

- **Diagnóstico Covid-19 positivo**, y confirmado por el protocolo centralizado, la aplicación enviará mensajes pseudoaleatorios anónimos con un **UUID (identificador único universal) hexadecimal** para casos positivos, y el formato ***mensaje_pseudoaleatorio***, a los dispositivos que se encuentren a 1 metro o menos. Estos mensajes no quedan almacenados en el dispositivo, ya que la comprobación para el resto de los usuarios es trivial.

- **Diagnóstico Covid-19 negativo**, la aplicación enviará mensajes pseudoaleatorios anónimos con un **UUID (identificador único universal) hexadecimal** para casos negativos, y el formato ***mensaje_pseudoaleatorio***. No obstante, se almacenarán los

mensajes enviados por el usuario, ya que cabe la posibilidad de contagio, falso negativo o asintomático.

- **Diagnóstico Covid-19 positivo a persona asintomática**, puede darse el caso que una persona haya estado usando la aplicación, y sin saberlo fuese asintomática positivo. Si dicha persona se realiza un diagnóstico de la enfermedad, siendo un resultado positivo, la aplicación realizará cambios automáticamente en los mensajes enviados en un rango de 10 días anteriores, para cambiar el formato almacenado, de ***nocovid:mensaje_pseudoaleatorio marca_de_tiempo*** a ***covid: mensaje_pseudoaleatorio marca_de_tiempo***. De esta forma, cambiarán los mensajes almacenados en el servidor en el protocolo centralizado, para que el resto de los usuarios que hubieran estado en contacto estrecho con este, sean notificados para realizarse el diagnóstico.

- **Cuando un usuario haya estado expuesto a un caso positivo**, la aplicación se lo notificará, y le sugerirá un diagnostico en la brevedad posible.

La estrategia mencionada anteriormente, se considera apropiada ya que cumple con los siguientes objetivos:

- El usuario puede instalar la aplicación en su dispositivo móvil siempre que lo desee, y éste sea compatible con las funciones requeridas.

- No requiere conexión entre dispositivos, evitando posibles fallos de seguridad en funciones determinadas, que pudieran aprovecharse por terceros de forma malintencionada para extraer datos privados de los usuarios.

- El protocolo descentralizado, facilita la determinación, si una persona contagiada por Covid-19 ha estado en contacto próximo con otras personas.

- El protocolo centralizado, por una parte, facilita la privacidad y veracidad de diagnóstico del usuario, y por otra, si un usuario contrae la enfermedad por contagio posteriormente, habiendo notificado un diagnóstico negativo inicial en la aplicación.

- El diagnóstico Covid-19 del usuario es privado, en ningún momento se utiliza información que identifique al usuario.

- El intercambio de mensajes entre dispositivos es totalmente anónimo.

- La verificación contra el servidor de mensajes (enviados) almacenados, igual que el punto anterior, no se comparte ni se

almacena información sensible, por tanto, no da lugar a la identificación del usuario.

- Si se produce una notificación al usuario de una posible exposición a la enfermedad, no se envía ningún dato personal a ninguna entidad. Se confía en la responsabilidad de la persona para efectuar las acciones pertinentes al mensaje de la aplicación.

- El usuario puede desinstalar la aplicación a voluntad sin perjuicio alguno.

# 4. Planificación del Trabajo

Los recursos necesarios para realizar el proyecto se detallan a continuación:

**Hardware:**

- Servidor de verificación de código **Covid-19** y almacenamiento de mensajes enviados.

- Mínimo dos terminales móviles Android para pruebas.

**Software:**

- **Android Studio,** para el desarrollo “Front-end” o programación de la aplicación de usuario.

- **NodeJS**, verificación local “**Back-end**” contra servidor de **código Covid-19** de usuario, almacenamiento de mensajes enviados y verificación de nuevos infectados.

**Tareas a realizar:**

- Investigar sobre la proximidad necesaria para que se produzca contagio del Covid-19.

- Investigar sobre el tiempo necesario para superar la enfermedad.

- Desarrollo de la aplicación Covid Tracing:

    - Crear las interfaces de usuario.
    - Programar el “Front-end” o código funcional del usuario.
    - Programar el “Back-end” o código funcional del servidor para verificar el diagnóstico Covid-19.
    - Programar el “Back-End” o código funcional del servidor para almacenar los mensajes enviados por los usuarios, y verificar los mensajes recibidos.
- Pruebas en local de la aplicación.
- Corrección de fallos.
- Compilar versión final de la aplicación.
- Entrega memoria final.
- Vídeo presentación del proyecto.
- Entrega de la vídeo presentación.
- Defensa del proyecto.

A continuación, se detalla una planificación temporal aproximada, mediante diagrama de Gantt:

| Nombre | Fecha de inicio | Fecha de fin |
|---|---|---|
| Trabajo final de máster | 16/9/20 | 14/1/21 |
| Problema a resolver | 16/9/20 | 16/9/20 |
| Objetivos del trabajo | 17/9/20 | 17/9/20 |
| Enfoque y método | 18/9/20 | 28/9/20 |
| Planificación | 29/9/20 | 14/1/21 |
| Investigar sobre contagio Covid-19 | 29/9/20 | 9/10/20 |
| Investigar tiempo necesario superación enfermedad | 29/9/20 | 9/10/20 |
| Desarrollo de aplicación Covid Tracing | 12/10/20 | 30/12/20 |
| Crear interfícies de usuario | 12/10/20 | 27/10/20 |
| Entrega interfícies | 27/10/20 | 27/10/20 |
| Programación "Front-End" | 28/10/20 | 24/11/20 |
| Programación "Back-End" servidor de verificación Covid-19 | 25/11/20 | 10/12/20 |
| Programación "Back-End" servidor de almacenamiento de mensaj... | 11/12/20 | 18/12/20 |
| Pruebas aplicación en local | 21/12/20 | 25/12/20 |
| Corrección de fallos | 28/12/20 | 30/12/20 |
| Entrega memoria final | 29/12/20 | 29/12/20 |
| Vídeo presentación del proyecto | 30/12/20 | 4/1/21 |
| Entrega vídeo presentación | 5/1/21 | 5/1/21 |
| Preparación defensa del proyecto | 5/1/21 | 8/1/21 |
| Defensa del proyecto | 11/1/21 | 14/1/21 |
| Final defensa del proyecto | 15/1/21 | 15/1/21 |

# 5. Requisitos funcionales de la aplicación

En los siguientes apartados se detallan los requisitos funcionales de la aplicación cliente.

## 5.1. Ejecución de la aplicación en segundo plano

Un usuario de **Covid Tracing**, aparte de utilizar dicha aplicación, el sistema debe permitir usar otras aplicaciones que disponga o descargue en el dispositivo sin interrupciones. Por tanto, la ejecución de esta aplicación se realiza en segundo plano, o dicho de otra forma, el usuario puede hacer uso de cualquier aplicación que necesite, aunque **Covid Tracing** esté ejecutándose.

## 5.2. Ejecución modo ahorro de batería

La aplicación debe ejecutarse añadiendo una excepción de ejecución para sí misma, para que pueda hacer el uso necesario de la batería del dispositivo móvil, ya que, de lo contrario, el sistema Android optimiza los recursos desactivando / minimizando actividades no esenciales, entre ellas las funciones de los periféricos de red, quedando afectada la función descentralizada, y causando mal funcionamiento en el intercambio de mensajes por proximidad de dispositivos.

## 5.3. Permisos necesarios

A continuación, se detallan los permisos que deberán ser habilitados por el usuario.

### 5.3.1. Requerir permiso al usuario para activar Bluetooth

La aplicación le requiere al usuario activar el periférico Bluetooth en caso se encuentre desactivado. Es imprescindible activar Bluetooth, dado que el intercambio de mensajes se calcula de acuerdo con la proximidad y detección de dicho periférico.

### 5.3.2. Requerir permiso al usuario para activar localización aproximada (Coarse location)

La aplicación requiere al usuario activar la localización aproximada, ya que los mensajes entre dispositivos, se intercambiarán en función de la distancia de encuentro con otra persona (1 metro o menos).

### 5.3.3. Requerir permiso de activación GPS

Se ha podido comprobar que activar la localización GPS, no es necesario efectuarlo en algunas versiones de Android antiguas. No obstante, y para descartar problemas en la ejecución de la aplicación se requiere la activación por parte del usuario.

### 5.3.4. Requerir permiso para excepción de ahorro de batería

Se requiere permiso al usuario para añadir la excepción de uso de batería por parte de la aplicación, ya que, de lo contrario, las funciones del Bluetooth no funcionan correctamente.

### 5.3.5. Permiso rechazado por el usuario

Si alguno de los permisos requeridos anteriores no se consiente, la aplicación finaliza la ejecución automáticamente.

## 5.4. Interfaces de usuario

En base a los datos analizados y recopilados sobre el virus **SARS-CoV-2**, a continuación, se detallan las interfaces con las que los usuarios interactuarán (**Figuras 1,2,3,4**), una vez instalen la aplicación en su dispositivo.

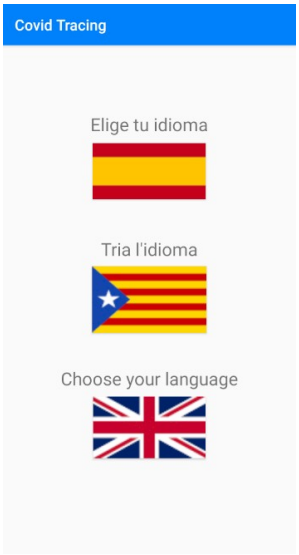



*Figura 1.* ***Interfaz de idioma***

Esta interfaz, permite a los usuarios elegir uno de los 3 idiomas disponibles con los que utilizar la aplicación.

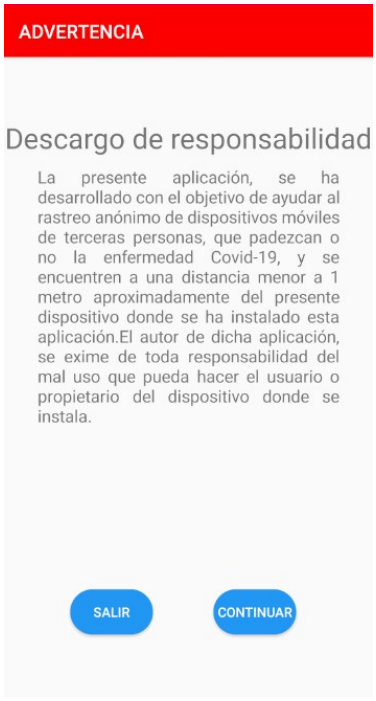



*Figura 2.* ***Interfaz de descargo de responsabilidad***

Donde se comunica al usuario la exención de responsabilidad del autor, en caso de hacer un mal uso de la aplicación. Se muestran 2 opciones:

- **Salir**, el usuario sale de la aplicación si no está conforme con la información proporcionada.

- **Continuar**, si está conforme con la información proporcionada, le llevará a la siguiente interfaz.

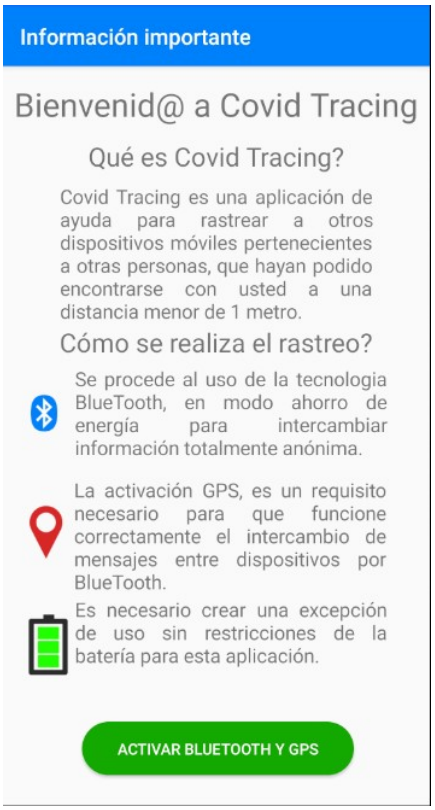



*Figura 3.* ***Interfaz de información y requisitos***

Informa a los usuarios de las funciones que requieren activarse para que la aplicación pueda funcionar correctamente. Solo que haya una función de las requeridas sin activar, la aplicación se cerrará automáticamente. Se muestra una opción:

- **Activar Bluetooth y GPS**, si el usuario activa esta opción, se le requerirá:

    - **Activar el dispositivo Bluetooth**, para el intercambio de mensajes anónimos entre dispositivos. Si no confirma la activación, sale de la aplicación.

    - **Ubicación aproximada**, requerida para el funcionamiento correcto de intercambio de mensajes entre dispositivos. Si no confirma la activación, sale de la aplicación.

    - **Activar localización GPS**, de momento es un requisito necesario para el intercambio de mensajes entre dispositivos. Si no confirma la activación, sale de la aplicación.

- **Habilitar excepción de uso de batería**, requerida para que funcione correctamente el intercambio de mensajes del componente Bluetooth, puesto que sin la aplicar dicha excepción, el modo ahorro de batería aplica restricciones en el uso de las comunicaciones Bluetooth, impidiendo que funcione como se requiere.

Una vez activados todos los requisitos, se accede a la siguiente interfaz.

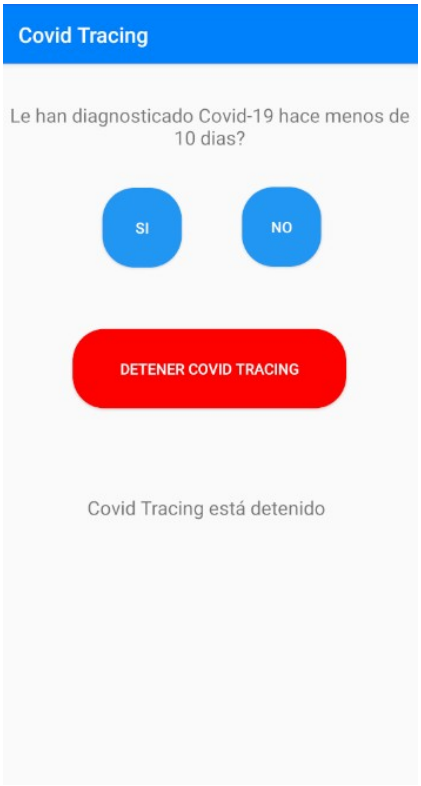



*Figura 4.* ***Interfaz de verificación y ejecución***

Se pregunta al usuario si le han diagnosticado la enfermedad Covid-19:

- **Caso positivo (Si):** automáticamente se le requiere un código (aún por determinar), el cual introducirá manualmente para verificarlo en una base de datos, y así validar de forma fehaciente el diagnóstico positivo, cuyas acciones según validación serán las siguientes:

    - **Positivo confirmado:** la aplicación se ejecuta como usuario Covid-19 positivo.

    - **Positivo falso:** la aplicación no se ejecuta, y queda a la espera que el usuario presione sobre el botón (No).

Presionado el botón, la aplicación se ejecuta en modo usuario Covid-19 negativo.

- **Caso negativo (No):** la aplicación se ejecuta en modo usuario Covid-19 negativo.
- **Detener Covid Tracing**, se detienen los servicios de la aplicación, y esta queda a la espera de:
    - Presionar nuevamente sobre Si.
    - Presionar nuevamente sobre No.
    - Salir de la aplicación de forma voluntaria.

# 5.5. Seguridad

En los siguientes subapartados se detallan los aspectos técnicos referentes a la seguridad de la aplicación cliente.

## 5.5.1. Privacidad del usuario (Intercambio de mensajes entre dispositivos)

El intercambio de mensajes se realiza cuando se detecta un dispositivo a partir de α>=-96db, es decir, aproximadamente, a partir de 1 metro de distancia o menos. En ese momento, mediante periodos de 25 segundos empieza el intercambio de mensajes, dentro de los cuales se envían entre 14 y 15 paquetes de información de la siguiente forma:

- Se genera un mensaje **pseudo-aleatorio** de 8 bytes.
- Se genera un **MAC (Message Authentication Code)** con una **función resumen SHA de 256 bits**, combinando el mensaje **pseudo-aleatorio** y una **clave privada de 1024 bits**.
- El resultado anterior, se codifica en **base64** truncando la salida a 30 bytes.
- Los 30 bytes truncados, se dividen en 3 partes de 10 bytes cada una.
- En intervalos de 25 segundos, se envía el mensaje pseudo-aleatorio **(parte constante)** y el paquete de 10 bytes **(parte variable)** de código correspondiente de las 3 partes, cada una

a su debido tiempo. Por tanto, los primeros 3 minutos se envía el primer bloque, a partir del tercer minuto se envía el segundo, y en el minuto 4,5 se envía el último bloque cifrado.

- Pasados los 5 minutos, se valida el mensaje **pseudo-aleatorio** y los 3 bloques cifrados recibidos. Si la validación es correcta, significa que ha sido creado por la aplicación y se almacena el mensaje más la marca de tiempo en caso de ser **Covid negativo**. De lo contrario, significa que el mensaje ha sido manipulado, y este queda descartado.

Si pasados los 5 minutos, y validado el mensaje recibido, si este fuese **Covid positivo**, la aplicación se lo notifica al usuario en la interfaz de verificación.

El intercambio de mensajes se realiza sin transmitir información sensible, como por ejemplo, el nombre del dispositivo, dirección IP, dirección MAC y etc. Además, se establece que en la transmisión de mensajes no se acepten conexiones. Por otra parte, cada transmisión crea una dirección **MAC (Media Access Control)** aleatoria, que no corresponde con la dirección original del dispositivo, evitando así su identificación.

### 5.5.2. Contenido de cada mensaje

El contenido de cada paquete de datos durante la fase de intercambio es el siguiente:

- **Servicio UUID (Universally Unique Identifier):** 16 bits, si es **Covid-19 positivo**, el contenido es **0xFD64**. Caso contrario, **0xFD6F**.

- **Servicio UUID de datos:** mismo caso anterior.

- **Datos:**
  - **mensaje pseudo-aleatorio:** 8 bytes (parte constante).
  - **Bloque cifrado:** 10 bytes (parte variable).

- **RSSI (Received Signal Strength Indicator):** 1 byte para obtener la intensidad recibida de la señal.

### 5.5.3. Conexión cliente a servidor para verificación del diagnóstico de la enfermedad Covid-19

Esta conexión es fundamental para garantizar la veracidad del diagnóstico del usuario, y por tanto, se establecerse una comunicación cifrada cliente-servidor mediante **certificado digital**. Además, se utiliza una cabecera que sea denominado **custom web token** en la petición de envío de conexión que consta de lo siguiente:

- Dos mensajes aleatorios que se definen dentro de un conjunto máximo de 63 caracteres, los cuales, mediante una función aleatoria de selección, se definen ambos de forma distinta entre un rango de 100 a 256 caracteres.

- Dichos mensajes aleatorios, se cifran con una clave privada utilizando un código de mensaje de autenticación de función hash SHA 512.

- Mensajes aleatorios y resultado del cifrado se codifican en base 64.

- Finalmente, se concatenan mensajes aleatorios y resultado del cifrado codificados en base 64.

### 5.5.4. Conexión a servidor cada 5 minutos para envío de mensajes enviados por el dispositivo, y comprobación de cambios de mensajes recibidos

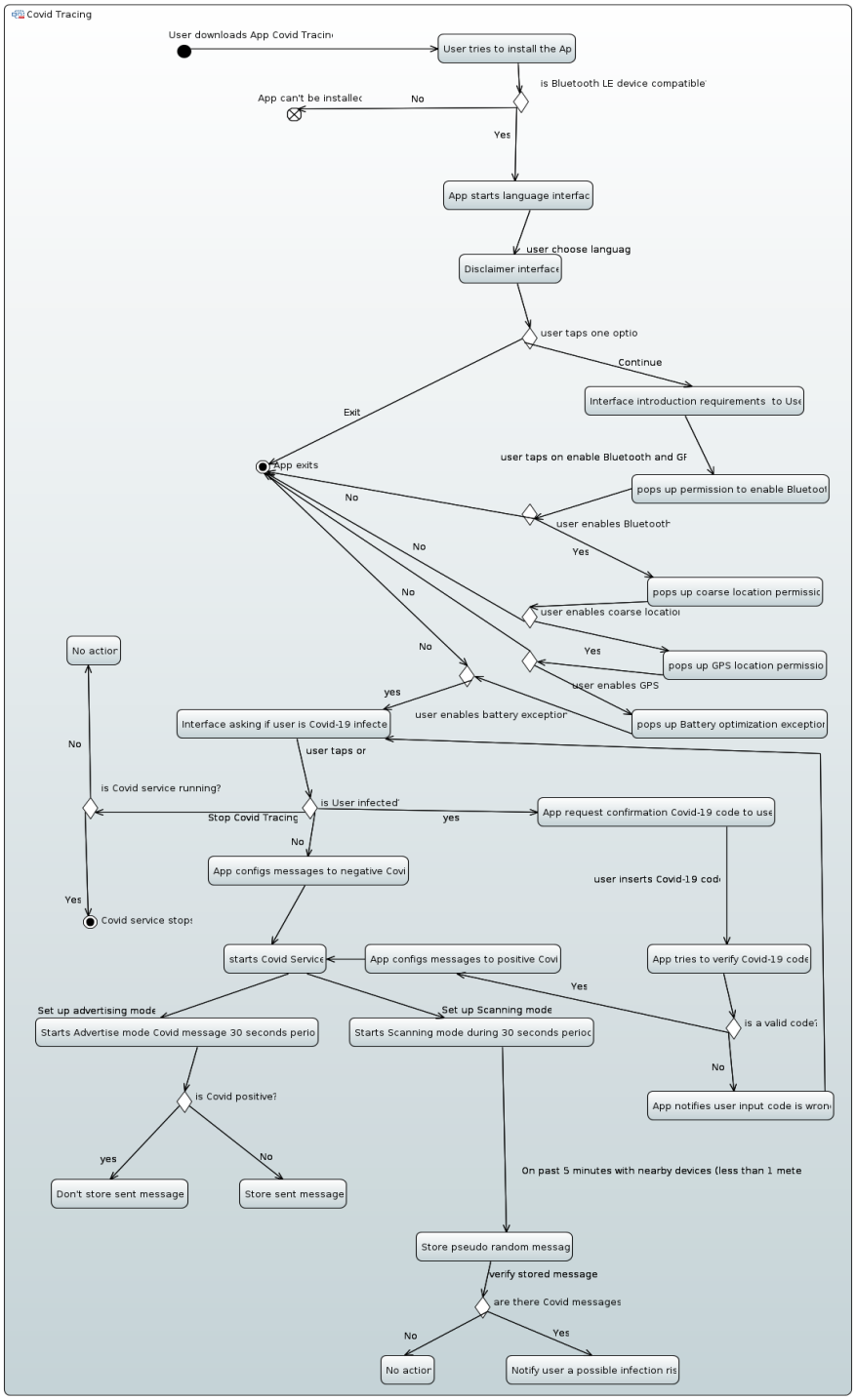
Mediante **okhttp3,** usando **certificado digital** y **custom web token** mencionado anteriormente, se realiza la conexión cada 5 minutos al servidor. La aplicación se conectará para volcar los mensajes enviados de forma descentralizada, y verificar los mensajes recibidos contra posibles actualizaciones de mensajes en el servidor, y notificar en caso de posible contagio.

### 5.5.5. Limpieza de mensajes almacenados en dispositivos móviles

Los mensajes almacenados en el dispositivo móvil tendrán una permanencia máxima de 15 días, salvo que se haya producido una posible notificación por contacto positivo, y el tiempo no sea superior a los 15 días establecidos de cuarentena con la fecha actual.

## 5.6. Diagrama de flujo de la aplicación

A continuación, se muestra el diagrama de flujo de la aplicación cliente:


Covid Tracing
User downloads App Covid Tracin
User tries to install the Ap
is Bluetooth LE device compatible
App can't be installe
No
Yes
App starts language interfac
user choose languag
Disclaimer interface
user taps one optio
Continue
Interface introduction requirements to Use
Exit
App exits
user taps on enable Bluetooth and GF
pops up permission to enable Bluetoot
No
user enables Bluetooth
Yes
pops up coarse location permissio
No
user enables coarse locatio
No
Yes
pops up GPS location permissio
user enables GPS
No actior
No
yes
user enables battery exceptior
pops up Battery optimization exceptior
Interface asking if user is Covid-19 infecte
No
user taps or
is Covid service running?
is User infected
Stop Covid Tracing
yes
App request confirmation Covid-19 code to use
No
App configs messages to negative Covi
user inserts Covid-19 cod
Yes
Covid service stop
starts Covid Service
App configs messages to positive Covi
App tries to verify Covid-19 code
Yes
Set up advertising mode
Set up Scanning mode
Starts Advertise mode Covid message 30 seconds perio
Starts Scanning mode during 30 seconds period
is a valid code
No
is Covid positive?
App notifies user input code is wron
yes
No
Don't store sent message
Store sent message
On past 5 minutes with nearby devices (less than 1 mete
Store pseudo random messag
verify stored message
are there Covid messages
No
Yes
No actior
Notify user a possible infection ris

## 5.7. Lógica del servidor (Back-End)

### Tipo de base de datos

Se utiliza una base de datos no relacional (NoSQL) orientada a objetos, MongoDB. Ofrece diversas prestaciones como, prestar servicio en local o en la nube, y puede prepararse para que sea una base de datos distribuida en ambos casos, en caso se requiera diversos servidores.

### Estructura de datos NoSQL

La estructura de datos puede basarse únicamente en 2 tablas, cuyos campos de almacenamiento sean tipo String (cadenas de caracteres):

- **patients**, tabla que almacena los identificativos covid de pacientes diagnosticados con la enfermedad como valor hash de función resumen.

- **randomsent**, tabla que almacena todos los mensajes que se van enviando de los dispositivos Android clientes.

# 6. Resultados obtenidos

### Entorno de ejecución

El entorno de ejecución de la aplicación **Covid Tracing**, se realiza en un ordenador portátil con sistema operativo anfitrión Linux, gestionando los recursos de la aplicación cliente 2 instancias del entorno de desarrollo **Android Studio**, a modo depuración, juntamente con dos dispositivos móviles.

La lógica del servidor “Back-End”, y por tanto de simulación, se gestiona mediante máquina virtual (VirtualBox) con sistema operativo Linux como invitado, utilizando la misma versión del sistema anfitrión.

### Aplicación cliente (Covid Tracing)

Una vez instalada y ejecutada la aplicación, esta se encarga de proporcionar la información para su uso, y los permisos que deben ser concedidos por el usuario para su correcta ejecución. Una vez realizada la configuración pertinente, la aplicación llevará a cabo las operaciones de transmisiones de datos entre los dispositivos cercanos a 1 metro o menos, conexiones a una máquina virtual, la cual desempeña la función

de simulación de un servidor, mediante certificado digital auto firmado para el envío y comprobaciones de datos.

## Servidor (VirtualBox) Node.js

Como se ha mencionado anteriormente, el servidor se ejecuta en un entorno Linux como sistema invitado, la aplicación se ha desarrollado con el framework Express (Node.js/JavaScript), y es responsable de realizar las siguientes funciones:

- Verificar la veracidad de diagnóstico Covid-19 positivo contra la base de datos.
- Recibir y almacenar los mensajes Covid negativos.
- Verificar pseudo mensajes aleatorios, y notificar a los usuarios en caso de contacto con un Covid-19 positivo.

## Servidor MongoDB

MongoDB es la base de datos no relacional (NoSQL), se encarga de comunicarse con el framework **Express (Node.js/JavaScript)** realizando las siguientes funciones:

- Almacena los identificadores Covid-19 positivos con una función resumen Hash.
- Almacena los mensajes Covid-19 negativos y marca de tiempo.
- Proporciona la información **covidid** para verificar un paciente Covid-19 positivo cuando se lo requiere el servidor **Express (Node.js/JavaScript).**
- Actualiza los mensajes de un paciente Covid-19 positivo cuando se le ha diagnosticado la enfermedad.
- Proporciona la información para la verificación, si algún paciente se le ha diagnosticado la enfermedad, estando inicialmente no contagiado cuando se lo requiere el servidor **Express (Node.js/JavaScript)**.

Los siguientes son los esquemas de las tablas / colecciones (figuras 5 y 6) que almacenan los datos en cada caso respectivo:

```
db.createCollection("patients", {
        validator:{
        $jsonSchema: {
                bsonType: "object",
                required: ["covidid"],
                properties:{
                        covidid:{
                                bsonType:"string"
                        }

                }
        }
 }
})
```

*Figura 5. Creación colección (patients) para almacenar los identificadores covid.*

```
db.createCollection("randomsent", {
        validator:{
        $jsonSchema: {
                bsonType: "object",
                required: ["randommessage","timestamp"],
                properties:{
                        randommessage:{
                                bsonType:"string"
                        },
         timestamp:{
                                bsonType:"string"
                        }

                }
        }
 }
})
```

*Figura 6. Creación de la colección (randomsent) para almacenamiento de mensajes enviados*

Los siguientes (figuras 7 y 8) son datos añadidos a las tablas / colecciones mostradas anteriormente, se muestran de forma respectiva:

```
> db.patients.find().pretty()
{
        "_id" : ObjectId("5fd38b85b2be8a9f8f83f1b5"),
        "covidid" : "c71ac32f2ad64ad9de5cfa81f45f73534243b082fe18f68a4597df33620
314d037f11ece97ef97fa5aa1fc8c7bf5edbae1daf82da8aea1b53cf081a826268235"
}
{
        "_id" : ObjectId("5fd38bcdb2be8a9f8f83f1b6"),
        "covidid" : "520b5edfba7710de67aa44a639c36299037bfe3f84fee371f20d4596adb
d62f488153fbdea092d94b5dd3410e2c3074b7fc8046835dcee058151d291d363df24"
}
>
```

*Figura 7. identificadores de pacientes insertados en la colección **patients***

```
> db.randomsent.find().pretty()
{
        "_id" : ObjectId("5fe3570b8f73510e1dd4c54a"),
        "randommessage" : "nocovid:uu54iUGT",
        "timestamp" : "23/11/2020"
}
{
        "_id" : ObjectId("5fe358558f73510e1dd4c54b"),
        "randommessage" : "nocovid:1H2Sxh4t",
        "timestamp" : "23/11/2020"
}
{
        "_id" : ObjectId("5fe3599f8f73510e1dd4c54c"),
        "randommessage" : "nocovid:f9Qzpy7R",
        "timestamp" : "23/11/2020"
}
{
        "_id" : ObjectId("5fe35aea8f73510e1dd4c54d"),
        "randommessage" : "nocovid:JUUNxtKO",
        "timestamp" : "23/11/2020"
}
{
        "_id" : ObjectId("5fe35c358f73510e1dd4c54e"),
        "randommessage" : "nocovid:ebXlzaHJ",
        "timestamp" : "23/11/2020"
}
{
        "_id" : ObjectId("5fe35d7e8f73510e1dd4c54f"),
        "randommessage" : "nocovid:s0uyTTGn",
        "timestamp" : "23/11/2020"
}
{
        "_id" : ObjectId("5fe367420a177c11c4d7662a"),
        "randommessage" : "nocovid:wBYVGLsQ",
        "timestamp" : "23/11/2020"
}
```

*Figura 8. mensajes pseudoaleatorios insertados en la colección **randomsent***

# 7. Conclusiones

## Lecciones aprendidas

En los siguientes subapartados se mencionan las lecciones aprendidas para las tareas fijadas en la planificación del trabajo.

## Investigación sobre la enfermedad Covid-19

Teniendo en cuenta que la enfermedad Covid-19, apareció de forma sorprendente y rápida, la recopilación de información y aprendizaje sobre la misma ha sido una tarea compleja, dado que casi a diario aparecían informaciones nuevas al respecto por los medios informativos, y debía contrastarse la veracidad con la que se manifestaban. Ha sido necesaria mucha paciencia y dedicación para llegar a las decisiones concretas que se han tomado en la implementación de la aplicación, a pesar de que siguen dándose nuevas noticias al respecto.

## Planificación del trabajo

La planificación del trabajo se ha realizado mediante un plan generalizado. No obstante, dada la complejidad del trabajo a realizar, se ha podido matizar lo suficiente para entender el desarrollo necesario para la funcionalidad conjunta, tanto de la aplicación cliente como la simulación con el servidor. Sin embargo, hubiera podido concretarse un poco más algunas de las tareas descritas, pero la novedad del proyecto y la falta de experiencia han influido sin duda.

## Aplicación cliente Covid Tracing

Se partía de un conocimiento muy básico del entorno de desarrollo **Android Studio**. A pesar de ello, se han aprendido tareas cómo:

- Estructurar el proyecto.
- Maquetación de las interfaces de usuario.
- Importar dependencias / librerías para su implementación.
- Definir permisos para la utilización de funciones de periféricos de un dispositivo móvil.
- Crear **certificado digital autofirmado [21]** mediante terminal de comandos con openssl.

- Conversión al estándar PKCS12 de un certificado digital.
- Crear un **par de llaves pública y privada [22]** mediante terminal de comandos con openssl.
- Definir archivos **gradle [23]** externos almacenados de forma interna para el sistema Android, en los que se almacena información necesaria para leer claves, y utilizarlas para el cifrado en el intercambio de información.
- Cargar información de archivos interna de la aplicación.
- Implementación de las funciones Bluetooth low energy.
- Funciones asíncronas.
- Crear servicios en segundo plano (**Background Service [24]**).
- Utilización de servicios con modo ahorro de batería activado.
- Comunicación entre un servicio ejecutándose en segundo plano y la intefaz de usuario.
- Conexión a servidor mediante certificado digital autofirmado y protocolo TLS.
- Aplicar **código de autenticación de mensaje [25]** en la cabecera de comunicación entre cliente y servidor.
- Aplicar **código de autenticación de mensaje** para intercambio anónimo de mensajes entre dispositivos móviles.
- Implementar **filtro de patrones [26]** para capturar mensajes anónimos

## Aplicación servidor (Back-End)

Para implementar la lógica del servidor, se partía desde nivel principiante en cuanto al lenguaje de programación JavaScript, utilizando el framework Express. Se ha aprendido lo siguiente:

- Instalar **Node.js [27]**.
- Preparar el proyecto mediante **Node.js [28]**.
- Instalación de módulos necesarios para el servidor.

- Implementar framework Express para Node.js.
- Conectar módulo de base de datos y servidor.
- Implementar archivos JavaScript de configuración para certificado digital y cifrado asimétrico.
- Implementar el protocolo HTTPS utilizando el certificado digital.
- Especificar IP y puerto de escucha para HTTPS.
- **Exportar funciones [29]** de módulos implementados en fichero JavaScript.
- Implementar middlewares para verificación de cabeceras.
- Función resumen con módulo externo para verificación de datos contra la base de datos.
- Ejecutar servidor para simulación en modo desarrollo.

## Base de datos MongoDB

Para la utilización básica de la base de datos **MongoDB**, ha sido necesario aprender los pasos que se mencionan a continuación:

- Instalar **MongoDB [30].**
- Acceder a **MongoDB** para la **creación de usuarios [31]** y otorgar privilegios de acceso, creación y modificación de colecciones y documentos pertinentes.
- **Crear las colecciones y tipos de datos [32]** para su almacenamiento.
- Implementar **control de acceso de autenticación y puesta en funcionamiento de MongoDB [33]**.

## Reflexión crítica sobre objetivos

Afortunadamente, y a pesar de la dificultad del proyecto, se han podido cumplir todos los objetivos impuestos. Sin embargo, se quería cumplir un objetivo fuera de planificación, el cual no ha sido posible, dada la complejidad que conlleva anonimizar de una forma eficiente, la dirección IP de los dispositivos móviles al realizar la conexión al

servidor. Aunque este no efectúa ningún tipo de almacenamiento respecto a este dato, pretendía conseguirse el anonimato absoluto del dispositivo en este aspecto.
En general, según una humilde perspectiva personal, probablemente se hayan cumplido más objetivos de los planteados en el proyecto, así que podemos concluir un cumplimiento satisfactorio.

## Análisis crítico de la planificación

Como ha podido insinuarse anteriormente, la temporización de la planificación no ha podido cumplirse de forma estricta, ya que se realizaban otras actividades diversas en paralelo al proyecto. Pese a no cumplirse estrictamente, no ha afectado a los tiempos establecidos para cada una de las entregas.

## Metodología prevista

A priori, se contemplaba una metodología más sencilla de la que se ha implementado. Este hecho, ha sido gracias a la implementación gradual de los mecanismos de comunicación y seguridad para anonimizar a los usuarios cifrando los datos, lo que ha causado observaciones importantes para la mejora de la aplicación durante el proceso de desarrollo, hecho que ha aportado una mejora considerable de la idea inicial.
Teniendo en cuenta que uno de los objetivos más importantes es mejorar la aplicación, e intentar que no tenga errores donde otras los comenten, era importante realizar las mejoras para satisfacer las necesidades observadas.

## Líneas de trabajo futuro

La principal línea de trabajo futuro se basa en la intención de investigar sobre los mecanismos existentes, o no, que puedan ser eficaces, a la par que eficientes, para esconder, enmascarar o anonimizar la dirección IP de los dispositivos móviles, en el momento de conexión al servidor correspondiente para el envío de datos, y recepción de notificaciones, ya que, desgraciadamente siempre existirán terceros que quieran saber más de lo que deberían. Por tanto, será interesante informarse debidamente al respecto, e intentar, aunque no exista una implementación completa, conseguir terminarla, o bien, a pesar sea costoso, documentarse para llevar a cabo una implementación novedosa y segura.

# 8. Glosario

Acrónimos utilizados:

- **Covid-19:** Corona virus disease 2019 (Enfermedad por Coronavirus 2019).
- **OMS:** Organización Mundial de la Salud.
- **QR:** Quick Response Code (Código de respuesta Rápida).
- **DP3T:** Decentralized Privacy-Preserving Proximity Tracing (Seguimiento de proximidad descentralizado que preserva la privacidad).
- **ID:** identificador.
- **API:** Application Program Interface (Interfaz de programación de aplicaciones).
- **BLE:** Bluetooth Low Energy (Energía de bajo consumo Bluetooth).
- **GATT:** Generic Attribute Profile (Perfil de atributos de generico).
- **UUID:** Universally Unique Identifier (Identificador único universal).
- **ATT:** Attribute Protocol (Protocolo de atributo).
- **HTTP:** Hipertext Transfer Protocol (Protocolo de transferencia de hipertexto).
- **JSON:** Java Script Object Notation (Notación de objeto JavaScript).
- **ORM:** Object Relational Mapping (Mapeo de objeto relacional).
- **MVC:** Model View Controller (Modelo Vista Controlador).
- **XML:** Extensible Markup Language (Lenguaje de marcado extensible).
- **DRY:** Don’t Repeat Yourself (no te repitas).
- **CSS:** Cascading Style Sheets (Hojas de estilo en cascadas).
- **HTML:** Hypertext Markup Language (Lenguaje de marcas de hipertexto).

- **Ipv6:** Internet Protocol version 6 (Protocolo de internet versión 6).
- **TLS:** Transport Layer Secure (Capa de transporte seguro).
- **SNI:** Server Name Indication (Indicación de servidor de nombre).
- **IDNA:** Internationalized Domain Name (nombre de dominio internacionalizado).
- **REST:** representational state transfer (transferencia de estado representacional).
- **PHP:** Hypertext Preprocessor (Pre-procesador de hipertexto).
- **SQL:** Structured Query Language (Lenguaje de consulta estructurada).
- **BSON:** Binary JSON (JSON binario).
- **GPS:** Global Positioning System (Sistema de posicionamiento global).
- **SARS-CoV-2:** severe acute respiratory syndrome coronavirus 2 (coronavirus de tipo 2 causante del síndrome respiratorio agudo severo).
- **MAC:** Message Authentication Code (Mensaje de código de autenticación).
- **SHA:** Secure Hash Algorithm (Algoritmo de hash seguro).
- **RSSI:** Received Signal Strength Indicator (Indicador de intensidad de señal recibido).

# 9. Bibliografía

Los siguientes han sido todos los enlaces consultados durante la realización del proyecto:

[1] OMS, *Organización Mundial de la Salud.* [En línea]. Disponible en: https://www.who.int/es/news/item/27-04-2020-who-timeline---covid-19 Accedido: 17/09/2020.

[2] El Español. *Coronavirus: La app de rastreo del COVID-19 será permanente en algunas ciudades chinas.* [En línea]. Disponible en: https://www.elespanol.com/omicrono/software/20200526/app-rastreo-covid-19-permanente-ciudades-chinas/492951138_0.html Accedido: 17/09/2020.

[3] Swissinfo. *Suiza lanza su app de rastreo SwissCovid - SWI swissinfo.ch.* [En línea]. Disponible en: https://www.swissinfo.ch/spa/respeto-a-la-privacidad_suiza-lanza-su-app-de-rastreo-swisscovid/45857368 Accedido: 17/09/2020.

[4] DP-3T. *GitHub - DP-3T/dp3t-app-android-ch: SwissCovid is the official contact tracing app of Switzerland.* [En línea]. Disponible en: https://github.com/DP-3T/dp3t-app-android-ch Accedido: 22/09/2020.

[5] TRONCOSO, Carmela, et al. *Decentralized Privacy-Preserving Proximity Tracing*. arXiv, 2020, p. arXiv: 2005.12273.

[6] TRONCOSO, Carmela, et al. *DP3T White Paper.pdf.* [En línea]. Disponible en: https://github.com/DP-3T/documents/blob/master/DP3T%20White%20Paper.pdf Accedido: 25/09/2020.

[7] XataxaAndroid. "*Radar Covid, análisis a fondo de su código: cómo funciona, qué está bien, qué está mal y qué falta.* [En línea]. Disponible en: https://www.xatakandroid.com/analisis/radar-covid-analisis-a-fondo-su-codigo-como-funciona-que-esta-bien-que-esta-mal-que-falta Accedido: 25/09/2020.

[8] Xataca. *"Nadie supo darme el código", el caos de Radar COVID: códigos que no llegan, notificaciones con retraso y mucho trabajo por hacer* [En línea]. Disponible en: https://www.xataka.com/aplicaciones/nadie-supo-darme-codigo-caos-radar-covid-codigos-que-no-llegan-notificaciones-retraso-mucho-trabajo-hacer Accedido: 27/09/2020

[9] Deutsche Welle. *"La aplicación alemana Corona Warn es un éxito, pero no para todos"* [En línea]. Disponible en: https://www.dw.com/es/la-aplicaci%C3%B3n-alemana-corona-warn-es-un-%C3%A9xito-pero-no-para-todos/a-53959705 Accedido: 28/09/2020.

[10] Android Developers. *Bluetooth low energy overview*. [En línea]. Disponible en: https://developer.android.com/guide/topics/connectivity/bluetooth-le Accedido: 28/10/2020.

[11] Bluetooth® Technology Website. *Topology Options*. [En línea]. Disponible en: https://www.bluetooth.com/learn-about-bluetooth/bluetooth-technology/topology-options/ Accedido: 03/11/2020.

[12] Android Developers. *Bluetooth overview.* [En línea]. Disponible en: https://developer.android.com/guide/topics/connectivity/bluetooth Accedido: 10/11/2020.

[13] INRIA. *On using Bluetooth-Low-Energy for contact tracing.* [En línea]. Disponible en: https://hal.inria.fr/hal-02878346v3/document Accedido: 10/11/2020.

[14] Square, Inc. *OkHttp.* [En línea]. Disponible en: https://square.github.io/okhttp/ Accedido:18/11/2020.

[16] Wikipedia *Node.js.* [En línea]. Disponible en: https://es.wikipedia.org/wiki/Node.js Accedido: 30/11/2020.

[16] Mozilla. *Frameworks Web de lado servidor - Aprende sobre desarrollo web*. [En línea]. Disponible en: https://developer.mozilla.org/es/docs/Learn/Server-side/Primeros_pasos/Web_frameworks Accedido: 05/12/2020.

[17] Wikipedia. *MySQL.* [En línea]. Disponible en: https://es.wikipedia.org/wiki/MySQL Accedido: 08/12/2020.

[18] Wikipedia. *Microsoft SQL Server*. [En línea]. Disponible en: https://es.wikipedia.org/wiki/Microsoft_SQL_Server Accedido: 11/12/2020.

[19] Wikipedia. *PostgreSQL.* [En línea]. Disponible en: https://es.wikipedia.org/wiki/PostgreSQL Accedido: 12/12/2020.

[20] Wikipedia. *MongoDB*. [En línea]. Disponible en: https://es.wikipedia.org/wiki/MongoDB Accedido: 15/12/2020.

[21] IBM. *Generating a self-signed certificate using OpenSSL*. [En línea]. Disponible en: https://www.ibm.com/support/knowledgecenter/SSMNED_5.0.0/com.ibm.apic.cmc.doc/task_apionprem_gernerate_self_signed_openSSL.html Accedido: 20/12/2020.

[22] Stack Overflow. *rsa - How to create public and private key with openssl?* [En línea]. Disponible en: https://stackoverflow.com/questions/44474516/how-to-create-public-and-private-key-with-openssl Accedido: 20/12/2020.

[23] Stack Overflow. *android - Is there a safe way to manage API keys?* [En línea]. Disponible en: https://stackoverflow.com/questions/33134031/is-there-a-safe-way-to-manage-api-keys Accedido: 22/12/2020.

[24] Android Developers. *Background Execution Limits.* [En línea]. Disponible en: https://developer.android.com/about/versions/oreo/background Accedido: 01/11/2020.

[25] Stack Overflow. *node.js - Compute HMAC sha-512 in NodeJS and Java.* [En línea]. Disponible en: https://stackoverflow.com/questions/54427887/compute-hmac-sha-512-in-nodejs-and-java Accedido:20/12/2020.

[26] Yosoy.dev. *Uso de expresiones regulares con el método Matches de Java*. [En línea]. Disponible en: https://yosoy.dev/expresiones-regulares-con-matches/ Accedido: 20/12/2020.

[27] Nodesource. *NodeSource Node.js Binary Distributions*. [En línea]. Disponible en:

https://github.com/nodesource/distributions/blob/master/README.md Accedido: 23/12/2020.

[28] TutorialsTeacher *Node.js Export Module.* [En línea]. Disponible en: https://www.tutorialsteacher.com/nodejs/nodejs-module-exports Accedido: 23/12/2020.

[29] Programarivm. *Pon en marcha un servidor https en Node.js con Express*. [En línea]. Disponible en: https://programarivm.com/pon-en-marcha-un-servidor-https-en-node-js-con-express Accedido:23/12/2020.

[30] MongoDB. *Install MongoDB — MongoDB Manual* [En línea]. Disponible en: https://docs.mongodb.com/manual/installation/ Accedido: 25/12/2020.

[31] MongoDB. *Users — MongoDB Manual*. [En línea]. Disponible en: https://docs.mongodb.com/manual/core/security-users/ Accedido: 25/12/2020.

[32] MongoDB. *Schema Validation — MongoDB Manual*. [En línea]. Disponible en: https://docs.mongodb.com/manual/core/schema-validation/ Accedido: 26/12/2020.

[33] MongoDB. *Enable Access Control — MongoDB Manual* [En línea]. Disponible en: https://docs.mongodb.com/manual/tutorial/enable-authentication/ Accedido: 26/12/2020.

# 10. Anexo

**Código aplicación cliente, versión no definitiva:**

https://github.com/jquerolmar/covidtracing/tree/background_dev

**Código aplicación servidor, versión no definitiva:**

https://github.com/jquerolmar/covidtracing/tree/backend_dev