PROYECTO FINAL DE INGENIERÍA

PROYECTO QRMe

Rodriguez, Andrés Santiago – LU1045512
Ingeniería en Informática
Panario, Franco – LU1046258
Ingeniería en Informática

Tutor:
De Armas, Adrián

2021

UNIVERSIDAD ARGENTINA DE LA EMPRESA
FACULTAD DE INGENIERÍA Y CIENCIAS EXACTAS

Índice

Índice de Imágenes

Índice de Tablas

1. Resumen

En la actualidad existen múltiples aplicaciones para conversar e intercambiar mensajes como WhatsApp o Telegram. Algunas poseen cifrado pero se debe activar manualmente, algunas ofrecen código QR pero solo uno por cuenta o se encuentra reservado para chats grupales, todas exigen número de celular para poder registrarse y todas exigen que la persona con la que quieres hablar también debe estar registrada en la aplicación.

Este proyecto de aplicación de mensajería plantea una alternativa para brindar al usuario la posibilidad de establecer un canal de comunicación cifrado y anónimo para ambas partes mediante la utilización y escaneo de un código QR, es decir, se crea una conversación que no requiere que la persona que escanea el código deba registrarse en la aplicación.

El usuario que genera el código puede además definir las reglas de conversación como el vencimiento del código, el horario en el que puede ser contactado, mensaje automático a mostrar y la cantidad de escaneos. El uso de esta aplicación permitirá a los usuarios poder ser contactados sin la necesidad de compartir ningún dato personal.

La misma está desarrollada en Ruby on Rails para la funcionalidad de BackEnd, mientras que para el FrontEnd utiliza HTML 5 con Ruby embebido y Bootstrap. La información se almacena en una base de datos MongoDB y el protocolo WebSocket lleva a cabo la funcionalidad de mensajería junto con Redis como base en memoria para la información intercambiada.

2. Abstract

Nowadays there are multiple applications to chat and exchange messages such as WhatsApp or Telegram. Some have encryption but it must be activated manually, some offer QR code but only one per account or it is reserved for group chats, all require a cell phone number to register and all require that the person you want to talk to must also be registered in the application.

This messaging application project proposes an alternative to offer the user the possibility of establishing an encrypted and anonymous communication channel for both parties through the use and scanning of a QR code, that is, a conversation is created that does not require that the person that scans the code must register in the App.

The user who generates the code can also define the conversation rules such as the expiration of the code, the time in which it can be contacted, automatic message to be displayed and the number of scans. The use of this Application will allow users to be contacted without the need to share any personal data.

The App is developed in Ruby on Rails for the BackEnd, while for the FrontEnd it uses HTML 5 with embedded Ruby and Bootstrap. The information is stored in a MongoDB database and the WebSocket protocol carries out the messaging functionality together with Redis as the in-memory base for the information exchanged.

3. Objetivo

Desarrollar un prototipo de aplicación móvil y web de mensajería instantánea segura que provea la posibilidad de establecer contacto de forma anónima a través de un código QR en Argentina en el año 2021.

4. Objetivos específicos

- Investigar y seleccionar las tecnologías y protocolos necesarios para llevar a cabo la aplicación.
- Definir e implementar herramientas para la organización de trabajo, incluyendo manejo y responsable de cada tarea, reuniones e investigaciones.
- Diseñar una aplicación que brinde un canal de comunicación anónimo y privado que se pueda generar a través de un código QR entre usuarios.
- Diseñar una base de datos no relacional para almacenar los distintos tipos de datos y estructuras.
- Desarrollar dentro de la aplicación un módulo de conversaciones que incluya protocolo de cifrado por defecto tanto para el intercambio de mensajes como el guardado de los mismos.
- Desarrollar e implementar dentro de la aplicación un módulo que permita generar códigos QR a partir de los cuales un usuario podrá escanearlo y generar una conversación sin estar registrado.

5. Alcance

En el primer release el producto estará compuesto por dos interfaces, una móvil resolviéndose a través de una aplicación de celular, y otra web. Dentro de las mismas un usuario podrá generar, exportar y administrar múltiples códigos QR, a través de los cuales un tercero podrá escanearlo y establecer contacto mediante mensajería instantánea. El usuario que genera dichos códigos QR podrá establecer reglas previas sobre el mismo, es decir, validez temporal del código, datos a mostrar una vez establecido el contacto con terceros, escaneos permitidos, respuesta automática y prioridad de notificaciones.

Además, en este release, tanto el usuario dueño de los códigos QR como el usuario qué lo escanea podrán posteriormente agregarse como contactos mutuamente teniendo la capacidad de mantener el anonimato. Estos nuevos contactos agregados estarán disponibles en una agenda de contactos. La conversación inicial quedará guardada en el archivo de conversaciones.

Este producto no incluirá grupos de conversación, agregar etiquetas a lista de contactos, integración con aplicaciones de entrega (por ejemplo: Rappi) o de pago (por ejemplo: MercadoPago), llamadas, videollamadas ni API para ser integrado con aplicaciones de terceros. Sin embargo, son funcionalidades que se agregaran en releases futuros.

6. Descripción

El avance en el mercado de productos como los pagos digitales y, más recientemente debido al COVID-19, el compartir información sin contacto, una mayor cantidad de personas se encuentra familiarizada con dicha tecnología y cómo utilizarla. Este contexto facilita la creación de un producto que permita establecer comunicación entre dos usuarios mediante el uso de dicha tecnología. [MEAÑOS, 2021]

Por otro lado, dentro del mundo de la mensajería instantánea, esta tecnología ya ha dado sus primeros pasos permitiendo que un usuario pueda conectarse con otro, como por ejemplo WhatsApp. Sin embargo, dicho servicio carece de la capacidad de mantener el anonimato de ambas partes ya que al hacer contacto escaneando el código QR, el número de teléfono se hace visible. Además carece de la posibilidad de administrar y personalizar dicho código de tal forma que permita establecer reglas de comunicación previas a que un tercero se contacte con la persona que genera el código. Por último, existe la limitación de poder crear un solo código, removiendo la posibilidad de tener varios con distintas finalidades con sus respectivas categorías y prioridades.

Entendiendo esto, se desarrollará una aplicación que permita establecer una comunicación a través de un canal cifrado y anónimo para ambas partes vía mensajería instantánea. Las características del chat en cuanto a tipos de mensajes permitidos y cantidad de información a brindar a terceros serán configuradas por el usuario qué desea ser contactado. Dentro de la aplicación, el usuario podrá generar múltiples códigos QR, los cuales se les podrán definir múltiples características: prioridad, categoría, información personal a brindar, aceptación de mensajes multimedia, vencimiento, escaneos permitidos, configuración de notificaciones y respuesta automática.

El usuario deberá poder descargarse la aplicación al celular de forma gratuita, dentro de la cual podrá registrarse sin la obligación de brindar un número de teléfono. Los usuarios que escanean el código deberán poder establecer conversaciones con el usuario dueño del código sin la necesidad de descargar la aplicación al celular, es decir, podrán intercambiar mensajes desde una ventana revolviéndose en un explorador web. Tanto como para escanear el QR y conversar con su

creador como para agregar terceros como contactos será condición necesaria que se encuentren registrados en la aplicación.

El usuario que desea ser contactado de forma anónima por cualquier tipo de motivo específico puede colocar el código QR dentro de un objeto que, en caso de ser extraviado, un tercero pueda escanearlo y comunicarse de forma directa, o en su puerta en caso de que esté esperando un paquete y necesiten notificarle la entrega, o en su vehículo en venta para qué posibles compradores puedan contactarlo, o colocar el código en la mochila de colegio de su hijo para ser contactado por cualquier cuestión de inmediato, entre muchos otros motivos que pueden ser relevantes.

7. Marco Teórico

En esta sección se describen los distintos conceptos y herramientas relacionados con el desarrollo de una herramienta con las características de una aplicación de mensajería que permita iniciar una conversación mediante el escaneo de un código QR.

7.1. Código QR

Es un tipo de código de barras que puede ser leído por un dispositivo digital y que almacena información como una serie de pixeles en una cuadricula de forma cuadrada. [KASPERKY, 2019]

El código QR brinda la ventaja de almacenar la información y obtenerla sin la necesidad de tipeo manual. Basta con hacer un escaneo del código para obtener dicha información. Además, agrega la capacidad de modificar la información que se desea mostrar de forma dinámica. Por el otro lado, es fácil de guardar y compartir, se pueden compartir como imágenes digitales o imprimir físicamente y la licencia es de libre uso y no requiere contrato.

Manteniendo el mismo código QR, se puede modificar la información presentada en el enlace asociado a dicho código sin la necesidad de generar uno nuevo o reimprimirlo. [SHUBHI MAL, 2016]

7.2. Privacidad y Seguridad

En esta sección se analizan los conceptos y tecnologías vinculadas a la capacidad de una aplicación de mensajería instantánea de brindar seguridad y privacidad sobre a los usuarios y los mensajes que intercambian.

7.2.1. ¿Qué es la privacidad digital y porqué es importante para el ser humano?

La privacidad digital consiste en el derecho de las personas a proteger sus datos en la red y poder decidir qué información dejar disponible o visible para el resto.

Cuando se habla de privacidad en la red, se refiere a todos los datos de la persona que se encuentran en internet. Esto incluye nombre, domicilio, DNI, teléfono, fotos, vídeos, localización, etc. [FERNANDO TABLADO, 2020]

La protección de estos datos es importante para el ser humano ya que permite crear barreras y protegerse de interferencias no deseadas, constituyendo además la base para definir la identidad de la persona.

Las personas que no protegen sus datos en la red o no poseen derechos de privacidad sobre sus datos son vulnerables a la intrusión de agentes externos. [PRIVACY INTERNATIONAL ORG, 2021]

7.2.2. Anonimato

Entendiendo el concepto de privacidad, en el ámbito digital el anonimato consiste en la capacidad de realizar acciones que permitan a uno ocultar su identidad, es decir, que no se pueda identificar a la persona. [SUDHANSHU CHAUHAN, 2015]

7.2.3. Cifrado en la comunicación

Dentro del ámbito de las comunicaciones existen diversos métodos para proteger la información intercambiada entre personas o que se mantenga inaccesible por parte de usuarios no autorizados. Uno de esos métodos se conoce como cifrado.

El cifrado es un proceso que consiste en que los datos que se intercambian se modifican desde el lado del emisor, convirtiéndolos en datos no legibles, para luego ser decodificados mediante una clave por parte del receptor.

En el contexto de una conversación, si se puede garantizar que el mensaje puede ser leído solamente por los participantes se le denomina “cifrado punta a punta”.

Los métodos para cifrar varían en tamaño y complejidad. De todas formas se pueden dividir en dos grandes grupos:

- **Simétrico**

  En un algoritmo simétrico tanto la clave de codificado como la de decodificado son idénticas.

  El mensaje a ser enviado es cifrado por una clave privada, luego transmitido, para finalmente ser descifrado usando la misma clave por el receptor. Presenta mayor riesgo ya que solo se necesita una clave para poder acceder a la información.

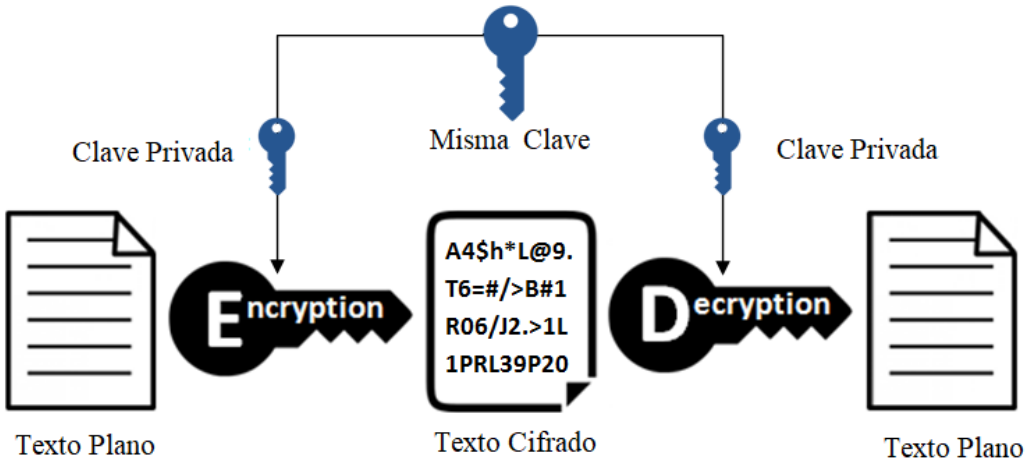


Imagen 1: Cifrado Simétrico [SSL2BUY, 2021]

- **Asimétrico**

En un algoritmo asimétrico se utilizan dos claves diferentes para el codificado y decodificado. Sin embargo dichas claves se encuentran relacionadas matemáticamente.

Una clave pública es utilizada para el emisor y una clave privada para el receptor.

Presenta un mayor nivel de seguridad ya que elimina el riesgo de tener que compartir la clave privada para poder acceder a la información pero su aplicación implica un mayor costo computacional. [KASPERSKY, 2021]

El proceso consiste en cifrar el texto con una clave pública que, luego de ser transmitido, sólo puede ser descifrado por el receptor con la clave privada.

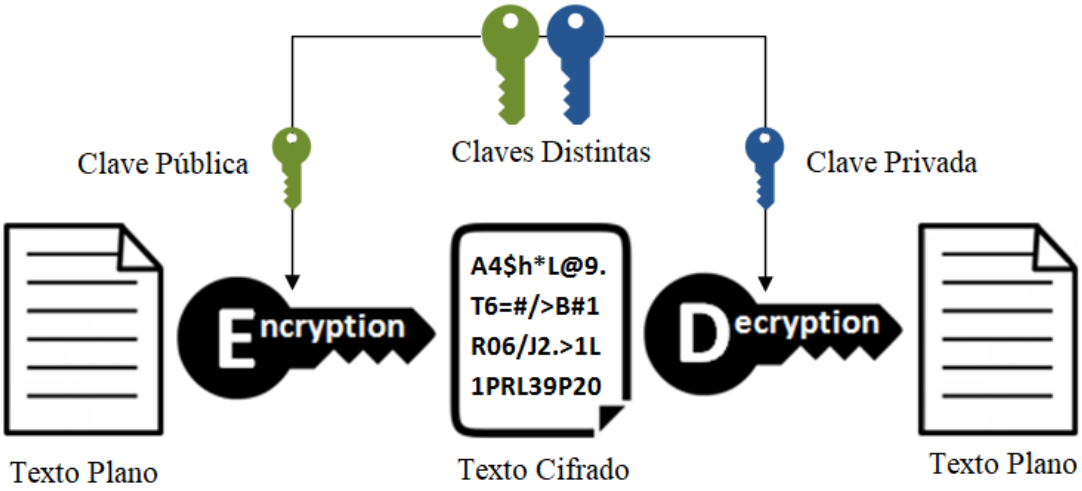


Imagen 2: Cifrado Asimétrico. [SSL2BUY, 2021]

7.2.4. Cifrado en una aplicación de mensajería

El cifrado en este tipo de aplicaciones consiste principalmente en que los mensajes intercambiados no puedan ser accedidos o leídos por personas externas o no deseadas, sea durante o posteriormente a la transmisión. Para ello existen diversos tipos de protocolos de intercambio de mensajes con sus respectivos niveles de protección. Los principales protocolos se analizan a continuación:

- **XMPP**

En inglés “eXtensible Message and Presence Protocol”, es un protocolo basado en XML enfocado en el intercambio de información estructurada con cifrado asincrónico. Si bien ofrece cifrado en el tráfico de mensajes, este no lo hace por defecto aunque permite agregarlo posteriormente

En un ambiente de intercambio XMPP, alrededor del setenta por ciento del tráfico lo concentra los datos de presencia lo cual dificulta su escalabilidad. Por otro lado, la arquitectura de sus redes es similar a la del correo electrónico y el intercambio de información se realiza mediante documentos cifrados en formato XML. [KSENIA ERMOSHINA Y OTROS, 2016]

- **OTR**

"Off-the-record" en inglés, es una extensión de XMPP creada para permitir cifrado punta-a-punta. OTR no posee claves públicas de largo plazo, es decir, se renuevan de forma constante. Si bien el protocolo ofrece cifrado punta a punta, solo funciona para el intercambio de mensajes síncrono entre dos personas y no permite mensajería en grupo o asincrónica.

- **Signal Protocol**

Basado en características de OTR, provee la posibilidad de intercambio asincrónico o grupal de mensajes. Las claves públicas se renuevan de forma constante y además oculta la identidad de los participantes, es decir, en caso de que una persona no deseada obtenga las claves, de todas formas no podrá identificar al emisor del mensaje.

El protocolo utiliza un generador de números aleatorios temporales para la creación de claves, el cual de volverse predecible compromete dicha generación. [CREMERS y OTROS, 2017]

- **OpenPGP**

El protocolo fue basado en el diseño PGP o "Pretty Good Privacy" en inglés, que fue originalmente diseñado para proveer cifrado punta-a-punta en correos electrónicos. Posteriormente el estándar fue abierto y dió origen a OpenPGP permitiendo implementaciones y aplicaciones del protocolo fuera del contexto de correos como, por ejemplo, cifrado de mensajes y directorios de contraseñas.

Si bien tuvo su origen en 1991, el impulsor de la estandarización de este protocolo IETF o "Internet Engineering Task Force" llevó a cabo un proceso de actualización para que utilice prácticas de cifrado modernas como, por ejemplo, mayor longitud en las claves. [KSENIA ERMOSHINA Y OTROS, 2016]

- **AES**

El estándar de cifrado avanzado o AES por su siglas en inglés, desarrollado por el Instituto Nacional de Estándares y Tecnología de Estados Unidos, es un método de cifrado simétrico

utilizado por las agencias de seguridad de dicho país para el cifrado de información clasificada además de ser utilizado por empresas como Bitlocker y Google.

AES funciona cifrando la información deseada con claves que pueden tener longitud de 128, 192 y 256 bits. La información a cifrar pasa por múltiples rondas en las que en cada una se sustituye, transpone y mezcla en conjunto con la clave dando como resultado el mensaje cifrado.

Al ser un método de cifrado simétrico, para aumentar la seguridad se implementa una arquitectura híbrida en la cual se realiza el cifrado con AES para aprovechar la mayor velocidad de este tipo de cifrados pero la clave generada se cifra con una clave privada antes de ser almacenada. [MICHAEL COBB y OTROS, 2021]

7.3. Protocolos de transferencia de datos

Habiendo analizado los posibles protocolos de cifrado, en esta sección se describen los protocolos de transferencia de datos relacionados con el desarrollo de una aplicación de mensajería.

Es necesario contar un protocolo de este tipo ya que define los mecanismos con los cuales la información será transmitida de un emisor a un receptor.

7.3.1. API

"Interfaz de programación de Aplicaciones" por sus siglas en inglés, una API es un conjunto de protocolos que permite realizar desarrollo e integraciones sobre aplicaciones de software.

Dichos protocolos habilitan la comunicación entre distintos servicios eliminando la necesidad de conocer cómo están implementados y estableciendo criterios para el intercambio de información. [REDHAT, 2021]

Dependiendo del requerimiento se pueden desarrollar cuatro tipos de API [STEPHANE CASTELLANI, 2020]:

- **Públicas**

Cualquier producto o servicio puede hacer uso de la API. Comúnmente no poseen restricciones de uso. Solo se necesita conocer los métodos de invocación de la misma.

- **De socios terceros**

Si bien no son de acceso abierto como una API pública, su existencia sigue siendo visible ya que suelen encontrarse descritas en los portales de API del servicio al que se desea acceder. Comúnmente se utiliza para brindar información o servicio fuera de la compañía.

- **Privadas**

Se encuentran escondidas de usuarios externos y solo se encuentra expuesta en portales internos de la compañía. Una API privada comúnmente se utiliza para integrar distintos desarrollos dentro de la misma compañía.

- **Compuestas**

Combina múltiples APIs y se construyen con el objetivo de permitir que el usuario pueda hacer llamados a múltiples servicios en simultáneo.

7.3.2. SOAP

Es un protocolo que se creó originalmente para permitir que aplicaciones con diferentes lenguajes y funcionando en diferentes plataformas puedan comunicarse. [REDHAT, 2021]

SOAP proporciona un mecanismo para intercambiar información estructurada en pares utilizando XML. El protocolo no define ninguna semántica, más bien define un mecanismo simple para expresar la semántica propia de la aplicación. [WORLD WIDE WEB CONSORTIUM, 2000]

Cuenta con reglas integradas que aumentan la complejidad y sobrecarga en los mensajes, característica propia del protocolo, y puede llegar a generar un tiempo de retraso en la respuesta del servidor.

Las especificaciones comunes de los servicios web incluyen:

- **Seguridad de los servicios web**

Protege y transfiere los mensajes utilizando identificadores únicos conocidos como tokens

- **Mensajería segura de los servicios web**

Controla los errores entre comunicaciones que se realicen en infraestructuras de TI poco confiables

- **Abordaje de los servicios web**

  Paquetes que enrutan, dentro de los encabezados SOAP, la información como metadatos

- **Lenguaje de descripción de los servicios web**

  Describe lo que realiza un servicio web desde su comienzo hasta el final de dicha acción

Los mensajes SOAP deben ser documentos XML, un lenguaje que pueden comprender tanto las personas como las máquinas. [REDHAT, 2021]. Estos documentos, se definen en tres partes en todos los mensajes:

- **Sobre**

Define una infraestructura que describe el mensaje y cómo se procesa. Dicho mensaje puede contar con cero o varias **cabezas** y un **cuerpo**. Las cabeceras transportan la información que sirve para el control del mensaje como puede ser atributos de calidad de servicio. El cuerpo contiene los parámetros del mensaje y su identificación. Y está formado por una colección de elementos.

- **Elemento**

Define la información que se incluye en el mensaje. Posee un namespace, en el cual se especifica en qué dirección se puede encontrar la definición del tipo de dato utilizado, el nombre del elemento y también un elemento puede contener sub-elementos en caso de que este sea de un tamaño considerable como grande o complejo y este formado por elementos más pequeños

- **Fault**

Es donde está contenida información sobre los errores que se puedan producir en el proceso de comunicación

7.3.3. REST

Define un conjunto de métodos de petición para identificar qué acción se desea realizar en base a una petición recibida. La información es entregada por medio del protocolo HTTP en estos posibles formatos: JSON, HTML, XLT, entre otros. [JSON ORG, 2021]

JSON es el más popular, debido a qué tanto las máquinas como las personas lo pueden entender rápidamente leyendo su convención tal cual es procesada por las máquinas Está basado en un subconjunto del lenguaje de programación JavaScript. [MOZILLA, 2021]

Está constituido por dos estructuras: una colección de pares de nombre/valor y una lista ordenada de valores. Estas son estructuras universales, todos los lenguajes de programación pueden soportarlo.

REST no es un protocolo ni un estándar, sino más bien es un conjunto de límites de arquitectura, es decir, los desarrolladores de las API pueden implementarlo de distintas maneras. [REDHAT, 2020]

7.3.4. Conexión a través de REST y SOAP

Tanto REST como SOAP son métodos que funcionan sobre el protocolo de petición/respuesta HTTP lo que significa que un cliente o usuario realiza una petición y obtiene una respuesta del servidor. En este tipo de intercambios el servidor no guarda el estado de las peticiones por lo que cada vez que termine el ciclo de petición/respuesta la nueva petición debe repetir información como, por ejemplo, el encabezado. [MARTIN FIETKIEWICZ, 2021]

Las conexiones a través de HTTP ofrecen seguridad en la capa de transporte agregando cifrado al intercambio de información, también conocido como HTTPS. [KUMAR CHANDRAKANT, 2020]

La siguiente imagen ilustra el intercambio a través de una conexión HTTP donde el usuario realiza una petición, recibe una respuesta, cierra la conexión y así sucesivamente por cada mensaje.

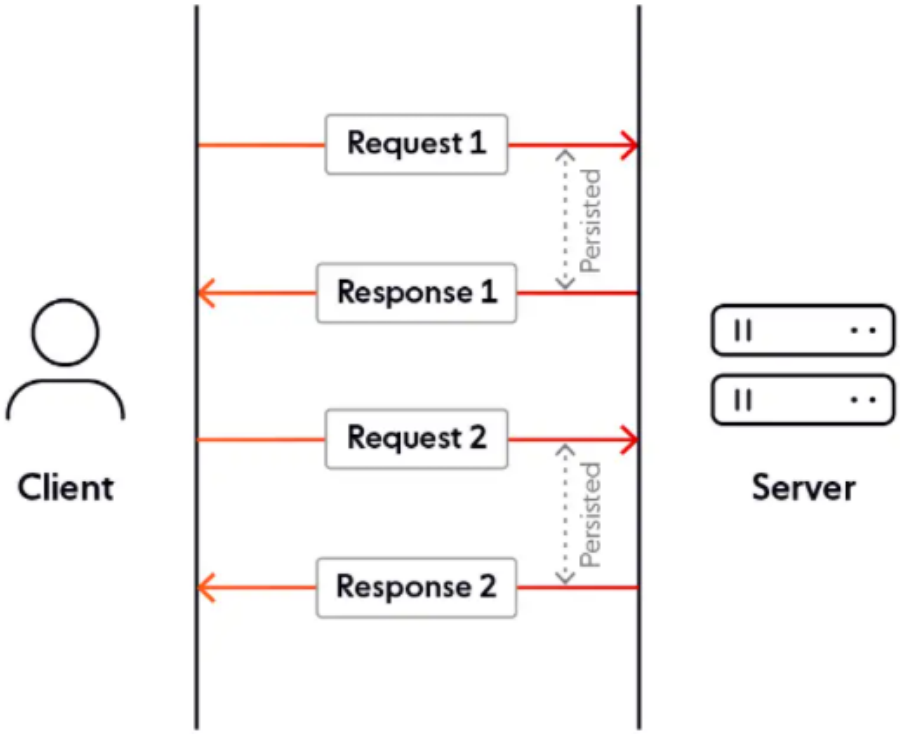


Imagen 3: Conexión HTTP. [MARTIN FIETKIEWICZ, 2021]

7.3.5. WebSocket

Una conexión a través de este protocolo comienza con una petición inicial entre el cliente y el servidor a través del protocolo HTTP, comúnmente denominado “apretón de manos” o “handshake” en inglés. Pero luego continúa el resto de la comunicación sobre el protocolo WebSocket a través de un canal full duplex, es decir, el cliente y el servidor pueden intercambiar información al mismo tiempo de forma independiente. Esta conexión se mantiene abierta durante toda la sesión y permite el intercambio de información en tiempo real. [MARTIN FIETKIEWICZ, 2021].

La imagen 4 ilustra el intercambio de mensajes a través de WebSocket. Se abre la conexión mediante HTTP, luego se mantiene la conexión abierta y se produce el intercambio mediante WebSocket y se cierra una vez finalizado.

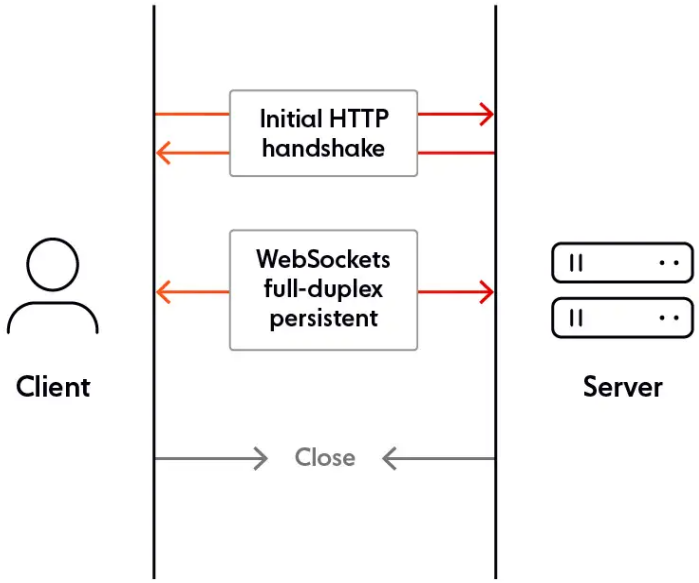


Imagen 4: Conexión WebSocket. [MARTIN FIETKIEWICZ, 2021]

Al igual que HTTP, ofrece seguridad en la capa de transporte o TLS (Transport Layer Security), al cual se le denomina WebSocket Secure o “WSS”. [KUMAR CHANDRAKANT, 2020]

Este protocolo permite cifrar los mensajes intercambiados reduciendo el riesgo de ataques como, por ejemplo, man-in-the-middle. [WEBSOCKET SECURITY, 2020]

7.3.6. Ejemplos de mensajes

Los métodos de intercambio de mensajes descritos en las secciones 7.3.2 “SOAP”, 7.3.3 “REST” y 7.3.5 “WebSockets” son los que han surgido en el estudio para elegir cual implementar en el desarrollo del proyecto. Dichos métodos se aplican para la mensajería de aplicaciones web. Se ha detallado las características principales y también, lenguajes y/o estructura de los mensajes de cada uno porque es un factor importante al momento de trabajar con un método u otro en consideraciones de velocidad, tiempo de respuesta, memoria y escalabilidad.

Las siguientes imágenes ilustran ejemplos de cada tipo de mensaje para entender la estructura y la forma que toman según se elija cada método de intercambio.

Ejemplo mensaje de REST

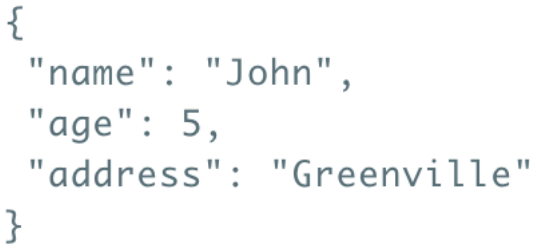


Imagen 5: Ejemplo REST

Ejemplo mensaje SOAP

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Header/>
  <soapenv:Body>
     <ns2:UserDetailsResponse xmlns:ns2="http://www.soapexample.com/xml/users">
        <ns2:User>
           <ns2:name>John</ns2:name>
           <ns2:age>5</ns2:age>
           <ns2:address>Greenville</ns2:address>
        </ns2:User
     </ns2:UserDetailsResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

Imagen 6: Ejemplo SOAP

Ejemplo mensaje WebSocket

- Establece la conexión o “handshake” mediante HTTP

```
GET /chat HTTP/1.1
Host: normal-website.com
Sec-WebSocket-Version: 13
Sec-WebSocket-Key: wDqumtseNBJdhkihL6PW7w==
Connection: keep-alive, Upgrade
Cookie: session=KOsEJNuflw4Rd9BDNrVmvwBF9rEijeE2
Upgrade: websocket
```

Imagen 7: Handshake HTTP. [PORT SWIGGER, 2021]

- El servidor acepta la conexión y actualiza el protocolo a WebSocket

```
HTTP/1.1 101 Switching Protocols
Connection: Upgrade
Upgrade: websocket
Sec-WebSocket-Accept: 0FFP+2nmNIf/h+4BP36k9uzrYGk=
```

Imagen 8: Upgrade WebSocket. [PORT SWIGGER, 2021]

- Una vez abierto el canal y el protocolo WebSocket establecido, los mensajes intercambiados pueden estar en muchos formatos. Un caso usando JSON puede ser:

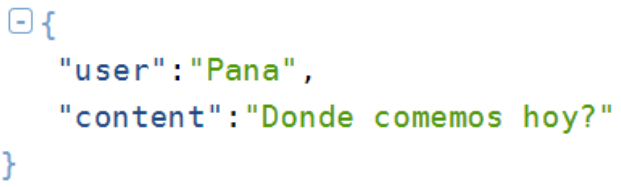


Imagen 9: Mensaje WebSocket

7.4. Base de datos

Dentro de la aplicación transitan y se generan conjuntos de datos cuya estructura cambia según la información que se almacene. Los mismos se generan en grandes volúmenes ya que cada interacción, mensaje y QR es almacenada.

Existen múltiples formas de almacenarlos, dentro de las cuales se encuentran las bases de datos relaciones y no relacionales, entre otras.

Teniendo en cuenta el tipo de desarrollo realizado, los datos son persistidos en una base de datos NoSQL siendo estas mejores para manejar grandes volúmenes de datos cuya estructura cambia de forma constante. [BENJAMIN ANDERSON, 2017]

A continuación se analizan ambos tipos de bases.

7.4.1. Base de datos Relacional o SQL

Grupo de elementos de datos con relaciones predefinidas entre ellos. Estos se organizan como un conjunto de tablas con columnas y filas. Dichas tablas se utilizan para guardar información sobre los objetos que se van a representar en la base de datos.

Cada fila de la tabla representa un conjunto de valores relacionados de una entidad y cada columna guarda un tipo determinado de datos. Un campo almacena el valor real de un atributo. [MICROSOFT, 2021]

7.4.2. Base de datos NoSQL

Tradicionalmente a lo largo de los años se han utilizado bases de datos SQL o relacionales para almacenar los datos. Estos se fraccionan en pequeñas porciones en formato de tabla con filas y columnas, y se relacionan a través de claves o identificadores.

A diferencia de estas, las bases de datos NoSQL o no-relacionales se denominan así por no poseer un identificador que sirva de clave para unir un conjunto de datos con otro. Por otro lado, no posee estructura fija, es decir, no requiere que se sepa de antemano que tipo de información se va a guardar ya que no posee esquema definido.

Al no requerir estructura predefinida, se pueden utilizar diversos tipos de estructura que optimice el tipo de dato que se desea guardar. Generalmente para almacenar información en este tipo de bases se usan pares de clave-valor, documentos JSON e incluso grafos con aristas y vértices. [AMAZON WEB SERVICES, 2021]

Estos tipos de base pueden ser documentales, clave-valor, orientadas a columnas u orientadas a grafos, siendo las orientadas a grafos diseñadas para almacenar las relaciones entre entidades por lo que no resultan útiles cuando no interesa persistir dichas relaciones. [Jennifer Reif, 2018]

¿Qué beneficios trae utilizar bases de datos NoSQL?

- **Escalabilidad**

Las bases de datos no-relacionales fueron creadas, entre otros objetivos, con la finalidad de poderse implementar con mayor facilidad en estructuras de escalamiento horizontal. En una arquitectura de este tipo la escalabilidad se logra distribuyendo o agregando más unidades de almacenamiento y procesamiento, lo cual presenta mayor dificultad si los datos están fuertemente relacionados como es el caso de las bases de datos relacionales.

Este atributo que poseen las bases NoSQL provee un camino más claro de escalabilidad cuando el volumen o el tráfico de datos aumenta. [AMAZON WEB SERVICES, 2021]

- **Datos estructurados versus datos no estructurados**

Las bases de datos relacionales requieren que, antes de cargar los datos, el esquema de las tablas se encuentre predefinido, es decir, el modelo debe estar diseñado y luego la información es transformada y cargada en dichas tablas. Cuando esta información es requerida, se recupera utilizando SQL y luego debe ser adaptada al formato de quien la necesita.

Las bases de datos NoSQL permiten que la información se almacene en múltiples formas o estructuras, lo cual genera que se requieran menos transformaciones por parte de quien necesita la información. Esta ventaja toma mayor importancia a medida que la cantidad de datos incrementa ya que a mayor cantidad de información, más costosas se vuelven las transformaciones. [MONGODB NOSQL, 2021]

Tipos de Base de datos NoSQL

Documentales

Es un tipo de base NoSQL cuyo funcionamiento se encuentra orientado a datos alojados en documentos o conjuntos de documentos.

Este tipo de bases almacenan todos los datos asociados y registros en un solo documento. Dentro del mismo documento los datos se encuentran semiestructurados lo cual agrega flexibilidad tanto para leer como para escribir información en la base. De esta forma se facilita el trabajo para los desarrolladores en caso de tener que modificar la estructura de los datos.

Una base de datos documental almacena la información en formato JSON, BSON o documentos XML.

Casos de uso incluyen plataformas de inversión en bolsa, aplicaciones móviles y plataformas de venta en línea. [MONGODB, 2021]

Algunos Productos son:

CouchDB

Imagen 10: Logo CouchDB

MongoDB

Imagen 11: Logo Mongo

Key-Value o Clave-Valor

Son bases de datos que utilizan el conjunto de pares clave-valor para el almacenamiento de datos. Dichas claves y valores pueden ser cualquier tipo de dato, desde objetos simples hasta complejos. Son altamente divisibles y poseen alta escalabilidad horizontal. En cierto sentido es como una base de datos relacional con dos columnas: la clave como atributo y el valor.

Casos de uso incluyen carros de compra en línea y perfiles de usuario. [GRAPH EVERYWHERE, 2021]

Algunos Productos son:

Apache Cassandra

Imagen 12: Logo Cassandra

Redis

Imagen 13: Logo Redis

Bases de datos orientadas a columnas

A diferencia de las bases de datos relacionales que almacenan la información fila por fila, los datos se organizan en conjunto de columnas, lo cual permite leer la información de solo las columnas deseadas sin consumir memoria en datos que no se requieren. Las columnas suelen ser del mismo tipo de dato lo cual mejora la compresión y hace la lectura más eficiente. Además las funciones de agregación sobre columnas son más rápidas lo que beneficia los trabajos de analítica.

Sin embargo, estas bases poseen mayor dificultad a la hora de escribir los datos ya que se requieren múltiples eventos de escritura para grabar una columna entera.

Casos de uso incluye almacenamiento para DataWarehouse y Big Data. [GRAPH EVERYWHERE, 2021]

Algunos Productos son:

MariaDB ColumnStore

Imagen 14: Logo MariaDB

Amazon Redshift

Imagen 15: Logo Amazon Redshift

7.4.3. Seguridad en bases SQL y NoSQL

Las bases de datos SQL al seguir un esquema de datos estructurado le permite tener soporte y control de acceso a nivel de fila y columna de cada tabla. Sin embargo dicho nivel de control de acceso puede afectar la velocidad de acceso a los datos

Por otra parte, al ser accedido mediante un lenguaje estructurado lo hace vulnerable a ataques como, por ejemplo, “Inyección SQL”. [MICHAEL COBB, 2021]

Por contraste, las bases de datos NoSQL al no accederse mediante lenguaje estructurado lo hace inmune a ataques del tipo inyecciones SQL, sin embargo este tipo de bases poseen su propia variante de ataques conocidas como inyecciones NoSQL, las cuales pueden venir en múltiples lenguajes como JavaScript y Python.

En ambos casos se puede reducir el riesgo de seguridad siguiendo buenas prácticas para el almacenado de datos sensibles:

- Cifrar la información antes de guardarla.
- Agregar validación al ingreso de datos.
- Aplicar mayores políticas de autenticación.

  [DAVEY WINDER, 2021]

7.5. Lenguajes y herramientas de desarrollo

Existen diversos lenguajes y herramientas de programación para desarrollar una aplicación de mensajería. En este contexto se analizan junto con conceptos que los envuelven.

7.5.1. Framework

Plataforma que proporciona una base para desarrollar aplicaciones de software. Utiliza recursos compartidos, como bibliotecas, archivos de imágenes y documentos de referencia, y los agrupa en un solo paquete. Ese paquete puede ser modificado para adaptarse a las necesidades específicas del proyecto.

Proporcionan una infraestructura que ya ha resuelto los detalles de bajo nivel, lo que permite centrarse en los detalles específicos del proyecto. Es decir, resuelve funcionalidades y/o configuraciones básicas de cierta/s característica/s del proyecto permitiendo enfocarse en la personalización y/o especificidad de la funcionalidad dentro de la aplicación. También se tiene que escribir menos código. Y menos código significa menos posibilidades de errores y menos tiempo de desarrollo. [CODE INSTITUTE, 2021]

7.5.2. Librería

Las librerías en lenguajes de programación son colecciones de código prescrito que se pueden utilizar para optimizar las tareas y funciones. [RYAN BARONE, 2020]

7.5.3. Aplicaciones Web

Las aplicaciones web son programas que utilizan modelo cliente-servidor, siendo el "cliente" el usuario que utiliza la aplicación. El "servidor" es la aplicación que administra las funciones y persiste la información.

Una aplicación web utiliza una combinación de distintos lenguajes de programación para manejar las funcionalidades del lado del cliente, como puede ser JavaScript, y lenguajes como Ruby o Python para las funcionalidades del lado del servidor. [INDEED EDITORIAL TEAM, 2021]

7.5.4. Bootstrap

Desarrollado por Twitter, Bootstrap es un Framework que combina CSS y JavaScript, principalmente utilizado para el desarrollo de la interfaz de usuario o front-end de una aplicación web.

Para poder manejar la interfaz, se provee una plantilla que contiene la definición de los elementos y estilos utilizados y agrega dos directorios. Uno con los archivos CSS para la

estilización de los elementos y otro con los archivos JavaScript, responsables de la ejecución de dichos elementos. [ROCK CONTENT, 2020]

La siguiente imagen ilustra un ejemplo de cómo es la estructura provista por una plantilla de Bootstrap, teniendo por un lado el directorio de JavaScript y por otro el de CSS.

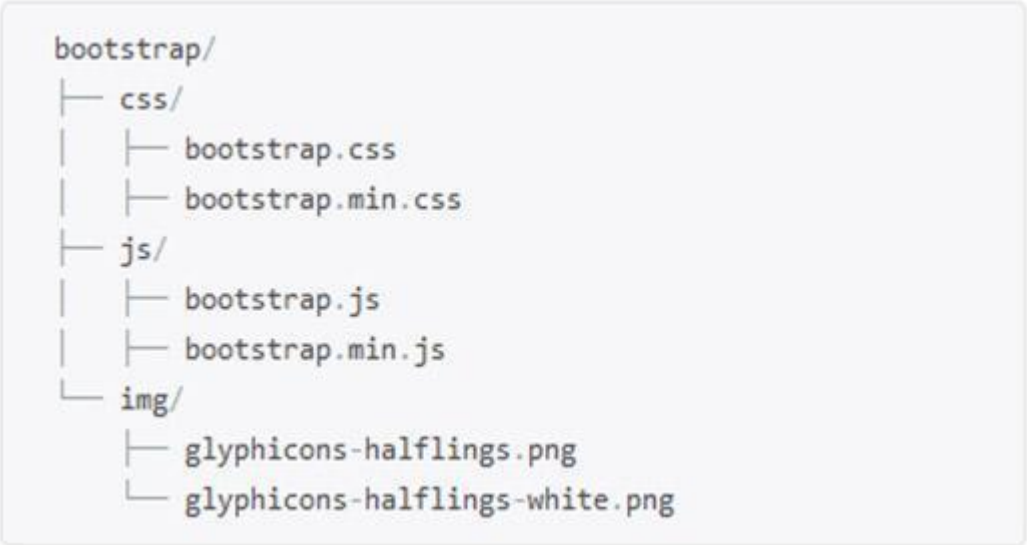


Imagen 16: Ejemplo Bootstrap

7.5.5. React

React es una librería de programación basada en JavaScript orientada al manejo de la interfaz de usuario o front-end, que a diferencia de Bootstrap, no utiliza plantillas sino que divide la interfaz de usuario en componentes donde cada uno maneja sus propios estados.[PETE HUNT, 2013]

A continuación se ilustra un ejemplo de estructura de React, el cual muestra cómo, a diferencia de Bootstrap, separa los archivos por componentes o funcionalidades juntando tanto los archivos CSS como los JavaScript en un mismo directorio.

```
common/
  Avatar.js
  Avatar.css
  APIUtils.js
  APIUtils.test.js
feed/
  index.js
  Feed.js
  Feed.css
  FeedStory.js
  FeedStory.test.js
  FeedAPI.js
profile/
  index.js
  Profile.js
  ProfileHeader.js
  ProfileHeader.css
  ProfileAPI.js
```

Imagen 17: Ejemplo React

7.5.6. Bootstrap vs React

Si bien ambos frameworks se encuentran orientados al desarrollo de la interfaz del usuario, es importante destacar las diferencias entre uno y otro para evaluar la conveniencia para QRMe.

- Bootstrap provee plantillas hechas con las características y estilos necesarios para que la aplicación sea responsiva. Dichas plantillas proveen una compatibilidad ya probada con CSS y JavaScript lo cual agrega mayor consistencia.

Además, la cantidad de temas y estilos es amplia y disponible gratuitamente en su gran mayoría.

De todas formas, se debe tener en cuenta que el hecho de que la plantilla ya se encuentre hecha puede hacer que el desarrollador deba invertir tiempo en aprender sobre los componentes y combinaciones incluidas.

Por último, los componentes que provee Bootstrap se encuentran listos para ser ejecutados ya que incluye todos los elementos para dicha ejecución. Todo esto permite ahorrar tiempo de desarrollo.

- React ofrece mayor performance en aplicaciones web que poseen gran cantidad de componentes ya que solo se actualiza el componente que lo requiere y no toda la página al mismo tiempo.

Sin embargo, React utiliza JSX que es una sintaxis que combina JavaScript y HTML, la cual posee mayor complejidad a nivel código.

Finalmente, analizándolo desde la perspectiva de “Modelo-Vista-Controlador”, React solo se enfoca en la vista por lo que requiere un mayor nivel de capas para poder desarrollar una aplicación completa. [HIREN DHADUK, 2021]

7.5.7. Ruby on Rails

Es un software de código abierto, siendo Ruby el lenguaje de programación y Rails el framework. Ruby es un lenguaje basado en el patrón MVC, interpretando todo como un objeto. Siendo el desarrollo de aplicaciones Web una de las áreas más populares en donde se utiliza este lenguaje.

Por el otro lado, Rails es un framework qué se destaca por su velocidad para la implementación de entornos de desarrollo, haciendo más ágil la posibilidad de visualizar los cambios realizados en el proyecto, la creación de todas las partes involucradas en un modelo de una aplicación orientada a objetos. Es decir, el directorio para las vistas, el controlador, el modelo en sí, la migración a la base de datos (dependiendo de la conexión qué se elija de la misma), test unitarios, etc. y la importación y utilización de librerías, algunas las mismas resuelven funcionalidades completas qué pueden necesitar en una aplicación. [RUBY, 2021]

7.5.8. Django

Django es un framework web basado en Python de alto nivel dedicado al desarrollo rápido y limpio en cuanto a su diseño. Orientándose, también, al desarrollo web y contando con la posibilidad de implementar librerías qué también resuelven aspectos generales de una aplicación como puede ser la lógica de creación y logueo de usuarios mediante cuenta de email contraseña. [DJANGO, 2021]

7.5.9. Ruby on Rails vs Django

Django al estar basado en Python, es recomendado para cuando una aplicación necesite características y funcionalidades centrales basadas en Machine Learning e Inteligencia Artificial.

Python en general falla en algunas tareas asíncronas, mientras que otros lenguajes sobresalen en eso. Twitch, por ejemplo, se destacó por el uso de Ruby on Rails y no habría funcionado bien si se hubiera creado con Django.

Por el otro lado se aconseja implementar proyecto en Ruby on Rails para cuando se necesita un armado de prototipo rápido y qué dicho prototipo sea un sitio o aplicación web. Es simple comenzar a implementar la lógica del proyecto y visualizar dicho trabajo además qué al momento de ser utilizado será simple su escalabilidad. [PRATHA AVASHIA, 2021]

7.5.10. Transmisión de mensajes

Para el intercambio de mensajes entre usuarios existen herramientas que facilitan el manejo del protocolo de conexión. Siendo que la aplicación requiere que los mensajes se transmitan en vivo, se evalúan librerías que operan sobre el protocolo WebSocket.

7.5.11. Action Cable

ActionCable es una librería integrada en el framework Rails que permite transmitir datos a través de WebSockets y administrar dichas conexiones.

La distribución de mensajes utiliza la metodología publish/subscribe. En otras palabras, los mensajes se publican en canales y los usuarios se suscriben a dichos canales. Esta información para ser transmitida en vivo se almacena temporalmente en una base de datos Redis y son eliminados de la misma una vez cerrado el canal de conversación.

Todos los canales de mensajes son administrados utilizando el protocolo WebSocket, el cual permite mantener la conexión abierta entre el cliente y el servidor el tiempo que requiera la aplicación. Luego, la librería crea las conexiones como objetos de Ruby lo cual facilita la integración con el resto del desarrollo incluyendo el manejo de autenticación de los usuarios dentro de cada canal de mensajes.

Por último, al utilizar WebSockets el intercambio de datos entre el cliente y el servidor es bidireccional por lo que ambos pueden recibir y enviar mensajes al mismo tiempo. [MATTHEW O'RIORDAN, 2020]

7.5.12. Socket.io

Es una librería desarrollada en JavaScript que permite realizar la comunicación bidireccional y basada en eventos entre el cliente y el servidor a través del protocolo WebSocket. La librería consta de dos APIs, una para el manejo de un servidor node.js y otra en JavaScript para el manejo de clientes.

Socket.io si bien se encuentra desarrollada nativamente en JavaScript, existen implementaciones en otros lenguajes como Java y Python mantenidas por la comunidad. [PETER KAYERE, 2020]

7.5.13. Action cable vs Socket.io

Ambas librerías utilizan WebSocket como protocolo para el manejo de conexiones e intercambio de datos y ambas aplican la arquitectura cliente-servidor. La principal diferencia yace en el lenguaje de desarrollo sobre cual es utilizado. Por ejemplo, si se desarrolla una aplicación en Ruby on Rail, se puede utilizar Action Cable ya que es nativo de dicha plataforma.

Las dos librerías permiten la autenticación de usuarios que intercambian los mensajes, no obstante, dichos mensajes pueden cifrarse para agregar otra capa de seguridad y aumentar la privacidad de dichos usuarios. [CHRIS HAUK, 2021]

8. Estado del Arte

Para explicar la situación actual del mercado en cuanto a aplicaciones de mensajería y elaborar un cuadro comparativo de servicios y funcionalidades se seleccionaron los productos que presentan mayor nivel de popularidad y nivel de usuarios activos, resultando en Facebook Messenger, WhatsApp, Telegram, WeChat y Line. Además, en dicho cuadro se evaluó la aplicación a desarrollar durante el proyecto final de ingeniería, QRMe. [STATISTA, 2021] [SIMILARWEB MESSAGING APPS, 2021]

El objetivo es resaltar y comparar las funcionalidades que agregaran un diferencial a QRMe del resto de las aplicaciones. Como se ha mencionado en la sección 6, “Descripción”, las funcionalidades que harán sobresalir a QRMe del resto están orientadas a la privacidad de datos y a la regulación de frecuencia y volumen de mensajes a recibir por parte del dueño del código QR.

8.1. Facebook Messenger

Imagen 18: Logo Facebook Messenger

Messenger, un servicio de mensajería instantánea de uso gratuito propiedad de Facebook, se lanzó en agosto de 2011 y reemplazó a Facebook Chat.

Se puede utilizar junto con Facebook en una computadora, en Messenger.com o accediendo a la aplicación móvil en dispositivos Android e iOS.

Facebook Messenger posee cifrado “end-to-end” para las conversaciones solo en sus versiones móviles. Las mismas no se encuentran cifradas por defecto sino que se debe seleccionar la opción “Conversación Secreta”.

Por otro lado, el desarrollo no es de código abierto.

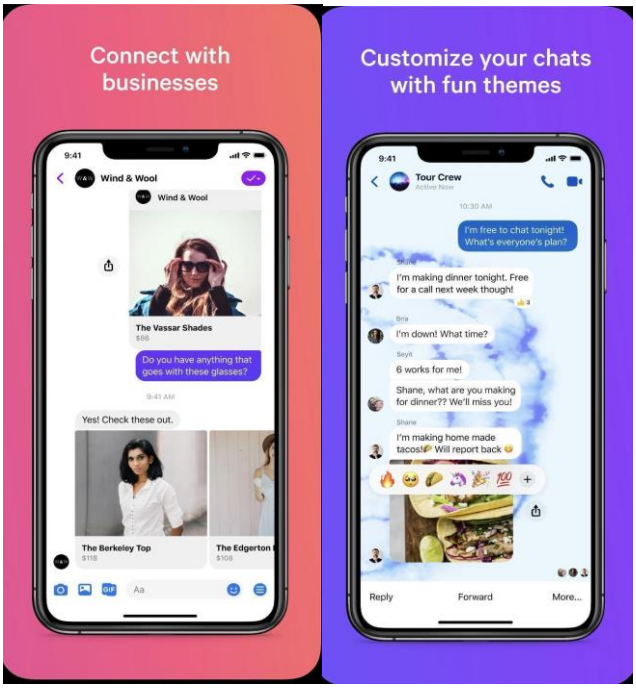


Imagen 19: UI Facebook Messenger

8.2. WhatsApp

Imagen 20: Logo WhatsApp

WhatsApp es una aplicación de mensajería instantánea lanzada en 2009 y posteriormente adquirida por Facebook en 2014. Es de uso gratuito y puede enviar mensajes, hacer llamadas de voz y realizar chats de video tanto en computadoras de escritorio como en dispositivos móviles.

Las conversaciones en WhatsApp poseen cifrado “end-to-end” por defecto. El desarrollo no es de código abierto.

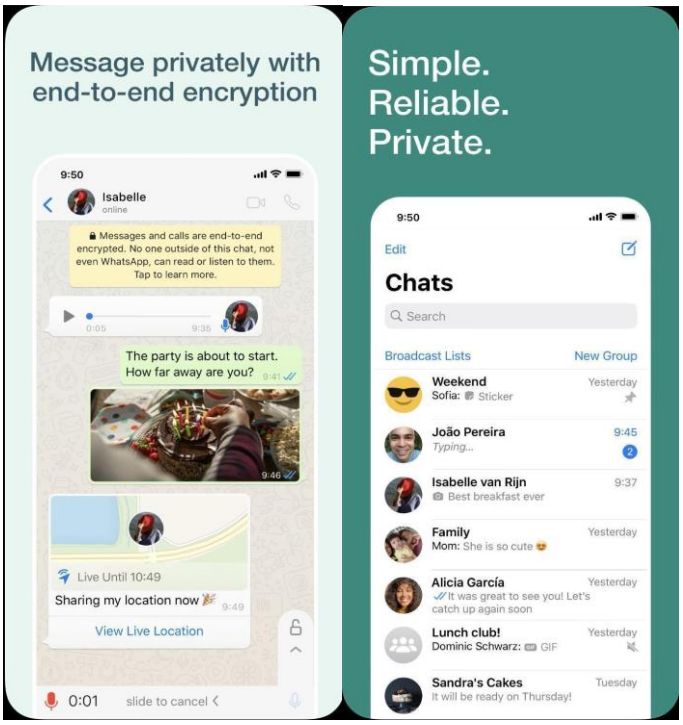


Imagen 21: UI WhatsApp

8.3. Telegram

Telegram es una aplicación de mensajería instantánea de texto y voz fundada por el empresario ruso de redes sociales Pavel Durov. El servicio es de uso gratuito y el desarrollo es de código abierto.

El servicio se encuentra disponible en iOS, Android y Web.

La aplicación ofrece cifrado “end-to-end” pero no viene activado por defecto. Se debe generar la conversación y luego habilitar la opción “Chat Secreto”.

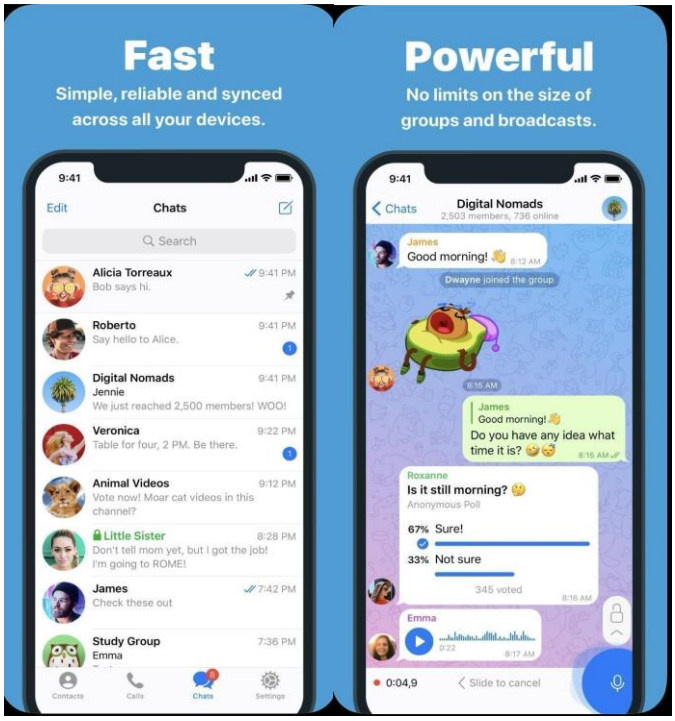


Imagen 22: UI Telegram

8.4. WeChat

WeChat es una aplicación de chat gratuita para dispositivos móviles de origen chino que ofrece videollamadas y llamadas de voz, videos, imágenes y mensajes de texto gratuitos entre otras cosas. Se encuentra disponible para iOS y Android pero no ofrece opción Web.

La aplicación no ofrece cifrado “end-to-end” ni tampoco desarrollo de código abierto.

Imagen 23: UI WeChat

8.5. Line

LINE es una aplicación de mensajería de origen japonés que ofrece mensajes, llamadas de voz y videollamadas gratis en cualquier momento y lugar. LINE está disponible en iOS y Android, y también ofrece aplicaciones para Windows y MacOS. Posee cifrado “end-to-end” pero no por defecto, debe activarse manualmente.

Las licencias y desarrollo de la aplicación no son de código abierto.

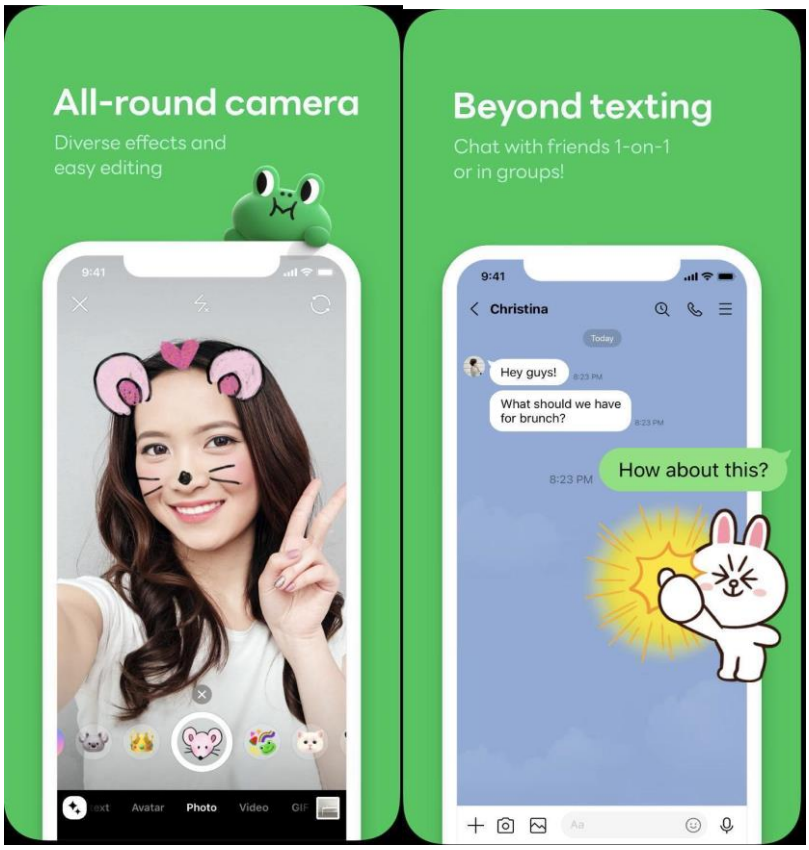


Imagen 24: UI Line

8.6. Diferencias entre aplicaciones

Tabla 1: Comparación entre aplicaciones

	Facebook Messenger	WhatsApp	Telegram	WeChat	Line	QRMe
Cifrado end-to-end	Si, en chat secretos	Si	Si, en chat secretos	No	Si, es opcional	Si
Requiere número de teléfono	Opcional	Si	Si	Si	Si	No
Conversación con usuarios no registrados	No	No	No	No	No	Si
QR único por conversación	No	No	No	No	No	Si
Deshabilitar conversaciones por fecha o número de escaneos del código	No	No	No	No	No	Si
Buscar Contactos	Por número, apodo o código QR	Por número	Por número, apodo o código QR	Por número , WeChat ID o código QR	Por número, Line ID o código QR	Apodo o código QR
Número de descargas	1,3 Mil Millones	2 Mil Millones	500 Millones	1,2 Mil Millones	167 Millones	No aplica

Analizando las aplicaciones y sus diferencias se observan distintos puntos de dolor que QRMe soluciona, destacando la capacidad de administrar la interacción con otras personas con códigos QR sin mostrar datos personales, el registro para utilizar la aplicación no requiere de número de teléfono ni ningún otro dato personal, el poder conversar con otra persona sin requerir que la misma esté registrada o tenga instalada la aplicación y el cifrado por defecto en todas las conversaciones.

9. Investigación de Usuario

9.1. User research: Encuesta

Se llevó a cabo una encuesta, adjunta en la sección 17.3 de Anexos, a 77 personas donde las preguntas apuntan a conocer las inquietudes, los usos y costumbres de los usuarios en relación a las funcionalidades y prestaciones de las aplicaciones de mensajería.

Por otro lado, la encuesta apunta a entender si existe un interés y/o preocupación sobre las problemáticas que QRMe busca resolver.

La encuesta hizo foco en mayores de 18 años que utilizan aplicaciones de mensajería.

Estos usuarios fueron consultados sobre qué aplicaciones de mensajería son las que más utilizan, donde WhatsApp resultó el claro dominante con un 98,7% de los encuestados eligiéndolo seguido muy atrás por Telegram, Otras y Facebook Messenger con un 24,7%, 23,4% y 19,5% respectivamente.

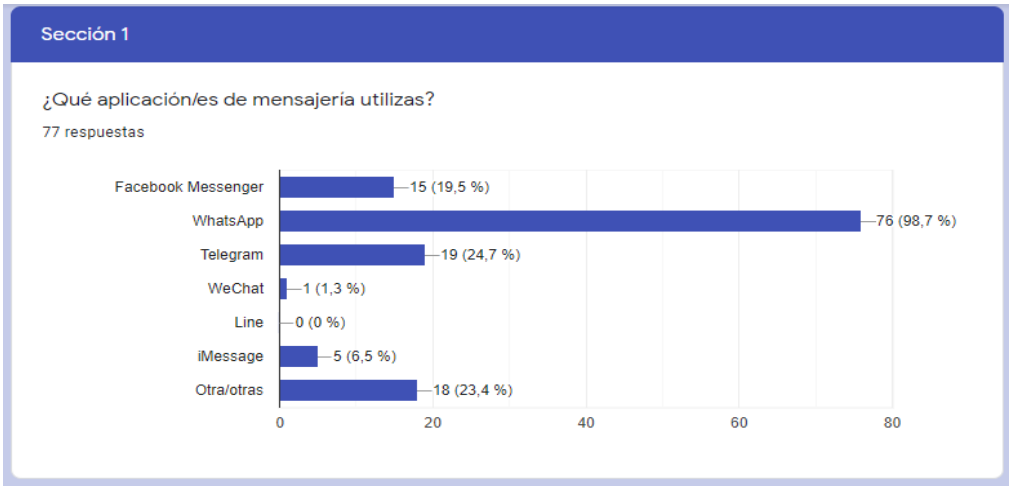


Imagen 25: Encuesta Estado del Arte 1

Sobre las aplicaciones consultadas se preguntó cuáles son los temas o motivos más importantes al momento de elegir cual utilizar. Siendo los consultados: entorno a la persona, gratuidad, anonimato y privacidad de los datos, funcionamiento, cifrado de las conversaciones y facilidad de uso, entre otras. Los más elegidos fueron que las personas del entorno de uno la utilicen

con un 90,9%, que sea gratis con un 76,6% y tercero el anonimato y privacidad de los datos con un 58,4%.

Dentro del marco de desarrollo de la aplicación QRME donde se considera como uno de los puntos más importantes a tener cuenta es la capacidad de la persona de poder mantenerse anónima al momento de comunicarse, más de la mitad de la población encuestada expresó que toma en cuenta dicha característica al momento de elegir que aplicación utilizar.

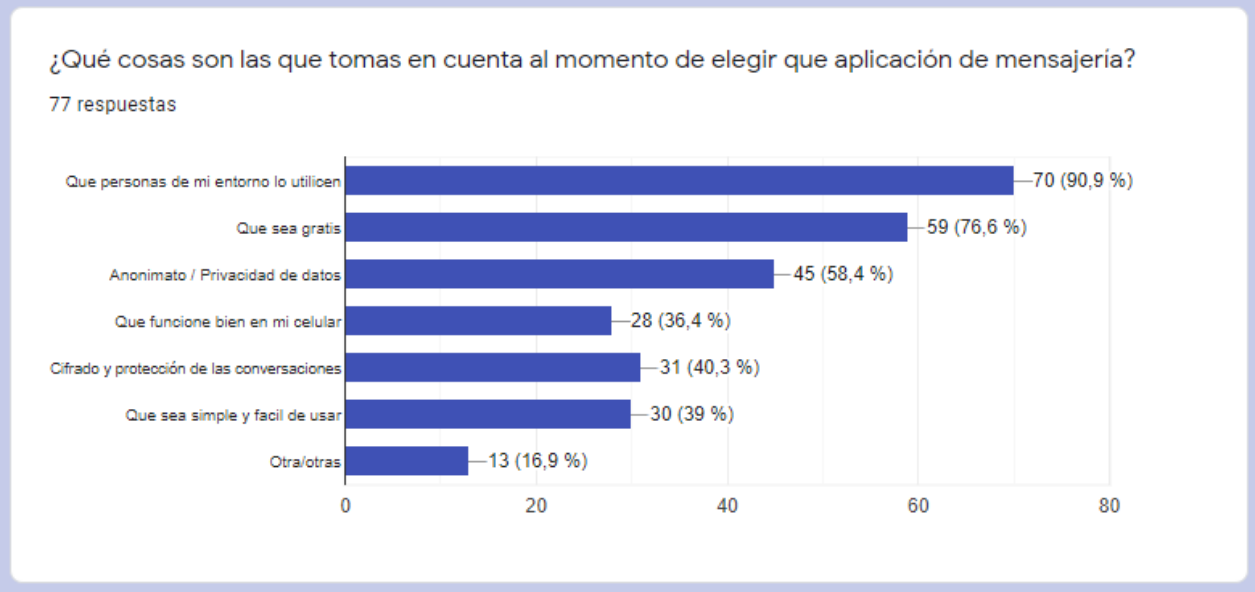


Imagen 26: Encuesta Estado del Arte 2

Por otra parte, interesa conocer de las aplicaciones de mensajería elegidas cuales son las funcionalidades que consideran más valiosas o importantes. Si bien las opciones más votadas fueron que se pueda compartir archivos multimedia con un 71,4% y poder tener grupos de conversación con varias personas con un 70,1%, un 66,2% eligió el poder iniciar la conversación de forma sencilla y un 44,2% voto por poder conectar con otras personas sin mostrar datos personales. Siendo estas últimas dos funcionalidades parte de lo que ofrece la aplicación QRMe.

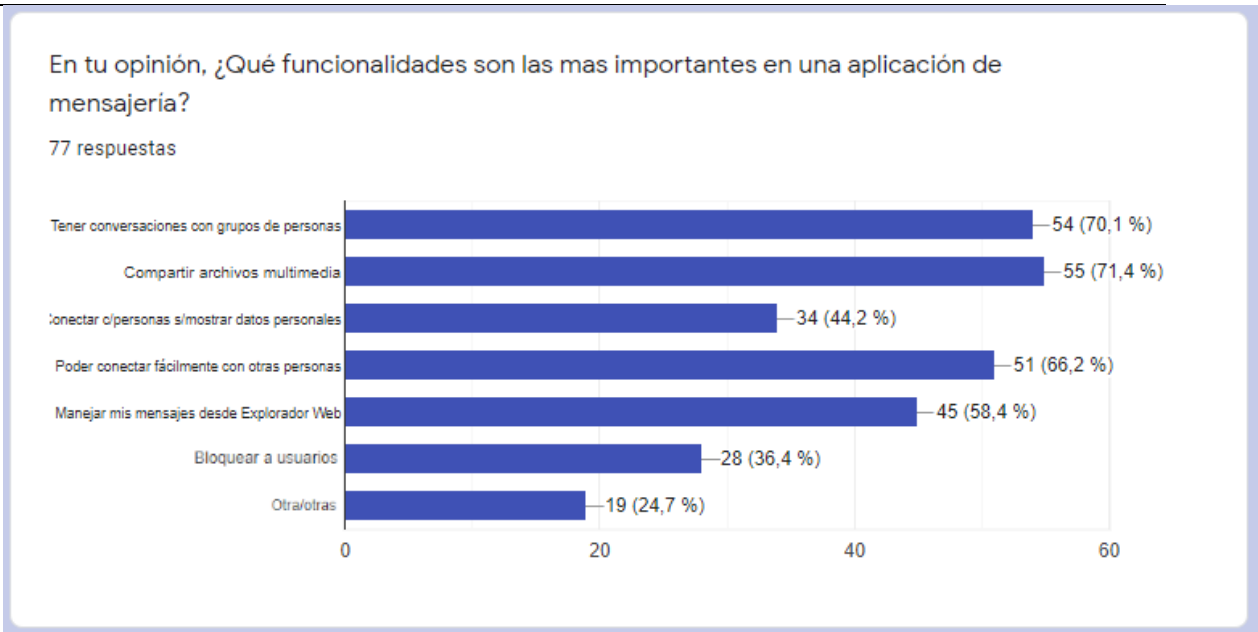


Imagen 27: Encuesta Estado del Arte 3

Por otra parte, se buscó conocer el interés por parte de los encuestados sobre varias funcionalidades que ofrece específicamente QRMe.

En primer lugar, se preguntó sobre el atractivo de poder definir múltiples reglas para cada conversación, donde más del 60% expresó tener interés en dicha funcionalidad.

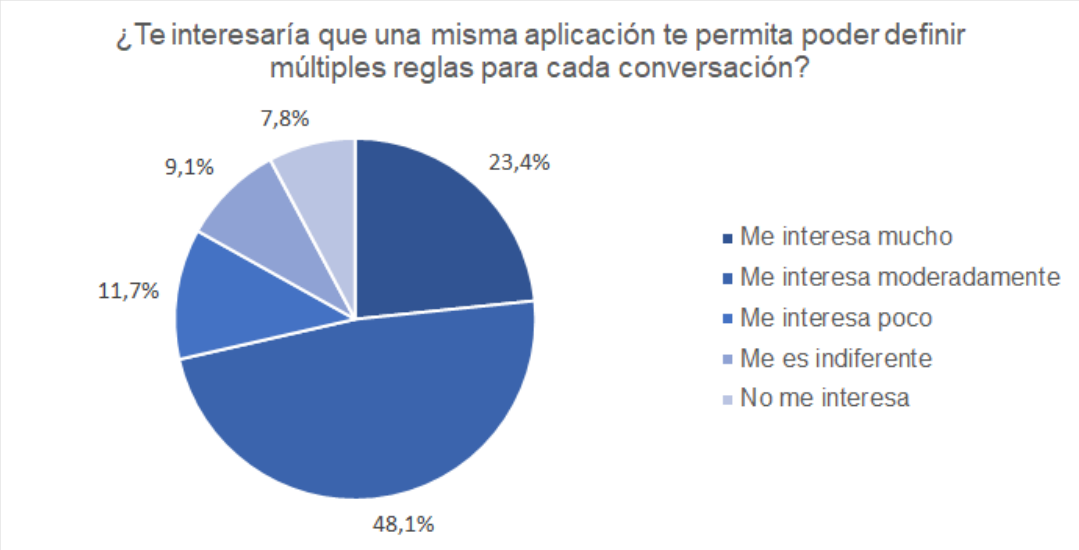


Imagen 28: Encuesta Estado del Arte 4

En segundo lugar, se desea saber si realmente los usuarios encuestados consideran un problema situaciones que las funcionalidades que ofrece QRMe planea mejorar y/o resolver. En este sentido se les preguntó sobre el caso hipotético en el cual deben proveer información personal, número de teléfono particularmente, con alguien que debería contactarlos por un compromiso específico y/o esporádico. Como resultado se obtuvo que más de un 70% manifiesta incomodidad en ese tipo de situaciones.

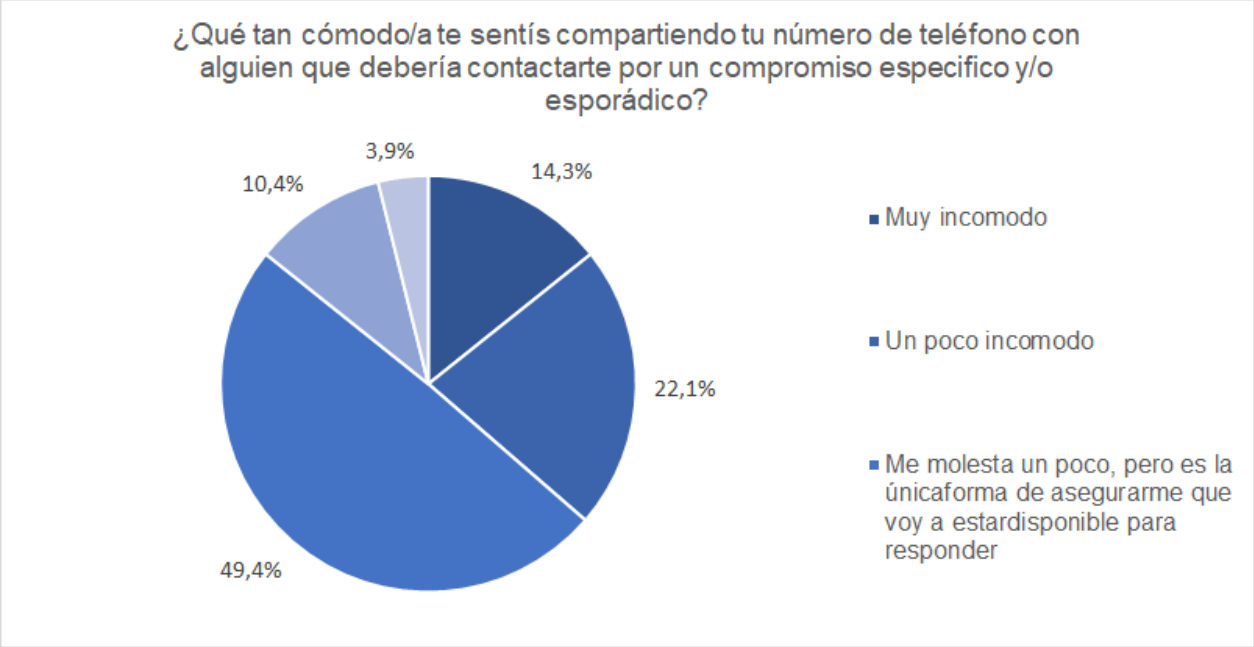


Imagen 29: Encuesta Estado del Arte 5

Siendo el código QR parte importante de la experiencia de usuario dentro de QRMe ya que es el medio por el cual los usuarios pueden conectar y conversar, se desea conocer el impacto por parte de los mismos de tener que utilizar dicha tecnología. Por lo cual fueron encuestados sobre su familiaridad con su existencia y uso.

En tal sentido, manifestaron en un 100% tener conocimiento o haber escuchado de la tecnología, con más de un 96% habiéndola utilizado por lo menos una vez.

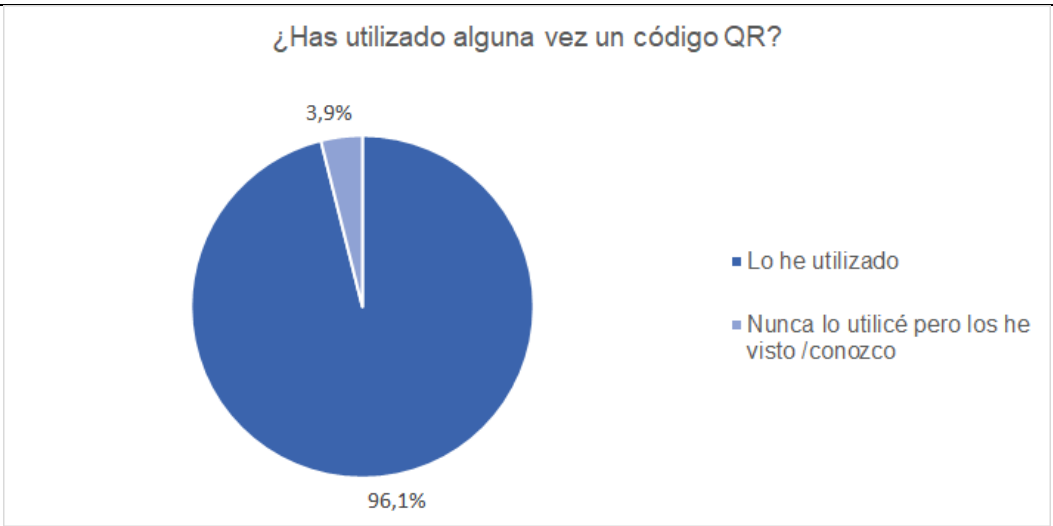


Imagen 30: Encuesta Estado del Arte 6

Por último, entendiendo el conocimiento de los encuestados sobre el código QR, se desea saber si existe interés en tener la posibilidad de conectar con otros usuarios sin la necesidad de tener que compartir número de teléfono ni ningún otro dato personal, sino a través del mencionado código QR. En este aspecto la encuesta reveló que más del 82% expresó interés en poder usar dicha funcionalidad.

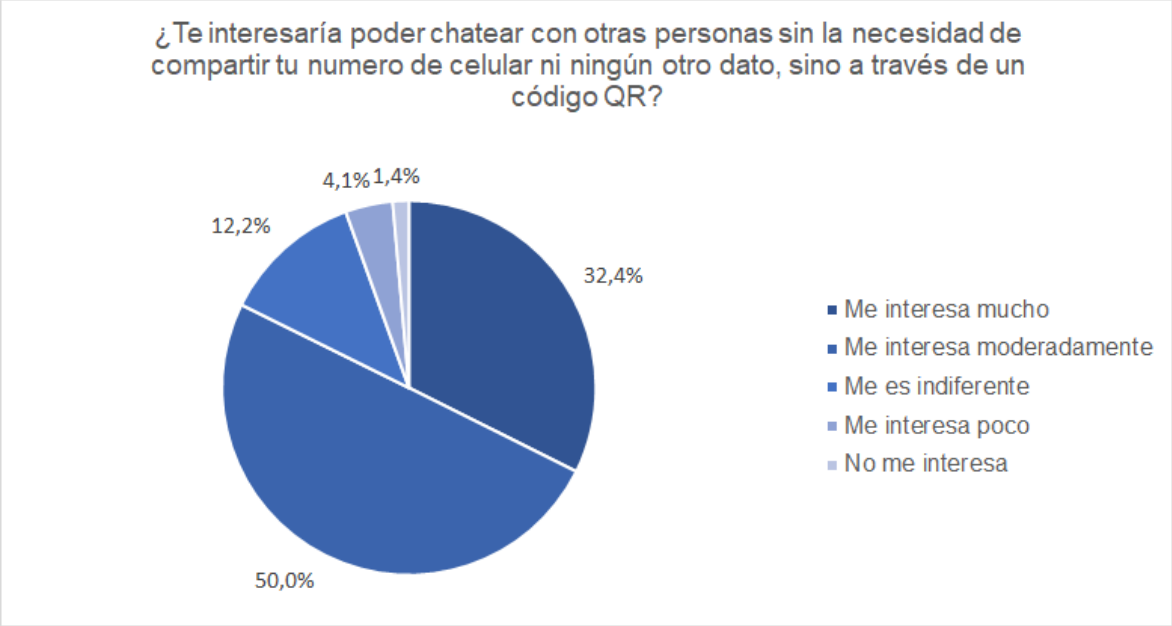


Imagen 31: Encuesta Estado del Arte 7

A partir de los resultados de la encuesta, se puede concluir que existe un interés en poder preservar el anonimato al momento de interactuar con otras personas mediante aplicaciones de mensajería.

Por otro lado, la encuesta reveló cierto nivel de conocimiento respecto al uso del código QR, dentro del cual manifestaron interés en tener la posibilidad de utilizarlo para poder conectarse con otras personas sin la necesidad de intercambiar datos personales.

Por otra parte, un alto porcentaje de los encuestados mostró interés por la funcionalidad de poder definir reglas específicas para cada conversación.

Finalmente debido a estos motivos, se puede concluir que existe un público que reconoce como problema las situaciones que QRMe pretende resolver y encuentra atractivas las funciones que el desarrollo planea ofrecer.

10. Desarrollo

A lo largo del desarrollo de la aplicación se tomaron decisiones de diseño de software, arquitectura de la aplicación, seguridad de la aplicación, programación, metodologías de trabajo y manejo de versión de código las cuales son descritas en las siguientes secciones.

Al momento de seleccionar el entorno en el cual debería funcionar la aplicación, se decidió ir por un desarrollo de aplicación web ya que brinda compatibilidad a través de todas las plataformas móviles y web, evitando la necesidad de desarrollo nativo para cada una.

En la sección 7.5, “Lenguajes y Herramientas” del Marco Teórico, se detallan todas las herramientas necesarias para el desarrollo de código de la aplicación, incluyendo la interfaz de usuario y el manejo y distribución de mensajes y códigos QR.

Por otra parte, en la sección 7.2.3 y 7.2.4, subsecciones de la sección 7.2 “Privacidad y Seguridad”, se detallan las dos clases de cifrado para el intercambio de mensaje que hay y los métodos de cifrado que se pueden aplicar sobre ellos. Para este trabajo se utiliza el esquema de cifrado simétrico AES.

Para el desarrollo de la aplicación, desde el lado del backend se utilizó el lenguaje Ruby y desde el lado del frontend JavaScript con HTML y CSS. En cuanto al framework de desarrollo, para Ruby se utilizó Ruby on Rails y para el frontend se implementó Bootstrap con JavaScript embebido dentro.

En la sección 7.3.5 del Marco Teórico se describe la tecnología WebSocket como protocolo de transferencia y en la sección 7.5.13 se describe también Action Cable como librería para administrar el protocolo WebSocket. Los motivos que impulsaron a seleccionar en el desarrollo estas tecnologías son, por un lado, WebSocket permite mantener un canal de conversación abierto en ambos sentidos que solo se cierra una vez finalizada la conversación reduciendo la carga y redundancia de información transmitida, y el por el otro, Action Cable al ser una librería nativa de Ruby para el manejo de canales mediante WebSocket, acelera y hace más sencilla la integración con el backend desarrollado en Ruby.

Para la organización de trabajo se buscó una metodología que permita la entrega y desarrollo constante de nuevas funcionalidades dentro de un equipo de trabajo. Se optó por adoptar de la metodología ágil SCRUM los componentes que mejor se adaptan a las necesidades del desarrollo.

Con respecto a la programación, se requirió de una herramienta que permita que más de una persona pueda trabajar en paralelo, que se pueda administrar que versión es productiva de la aplicación y cuáles no, y por último que permita el versionado y respaldo del código. En este trabajo se seleccionó Git como la herramienta mediante la interfaz de usuario provista por GitHub.

10.1. Selección de arquitectura

Por la naturaleza de este tipo de aplicaciones, la estructura de los mensajes puede cambiar constantemente según la información que se almacena. Además existe la posibilidad de requerir la información en tiempo real por lo que el tipo de base de datos que más se adapta es NoSQL seleccionando MongoDB como programa.

El servidor se monta en Puma Web Server, siendo este un servidor que permite múltiples llamadas en simultáneo, como puede ser el caso de recibir y enviar múltiples mensajes al mismo tiempo.

La conexión se realiza por HTTP o por WebSocket dependiendo de lo que se requiere transmitir. HTTP es utilizado para hacer el primer contacto entre el usuario y servidor para abrir el canal de conversación, luego para la transmisión de los mensajes se utiliza WebSocket ya que permite mantener la sesión abierta hasta que termina la conversación y además permite recibir y enviar mensajes al mismo tiempo entre el usuario y el servidor.

Los mensajes al ser creados necesitan un canal donde almacenarse temporalmente hasta ser transmitidos, a este proceso se le denomina “publicación”. Por otra parte, los usuarios se conectan a dichos canales para recibir los mensajes, a este proceso se le llama “suscripción”. Estos canales son almacenados y administrados en una base de datos temporal Redis, la cual una vez cerrada la transmisión por parte del WebSocket, los datos son eliminados de dicha base.

Todo el backend es administrado desde el framework de Rails, el cual se encuentra diseñado con el patrón de diseño de software “MVC” o “Modelo-Vista-Controlador” ya que este nos permite administrar múltiples vistas dependiendo si el usuario es el dueño del código QR o si es un usuario que escanea el QR para conversar. También brinda la capacidad de realizar cambios sobre alguna parte de la aplicación sin que esta impacte en el resto de las partes. Por último, facilita el desarrollo en paralelo ya que, por ejemplo, una persona puede estar desarrollando sobre el modelo mientras que otra persona trabaja sobre el controlador.

El diagrama de la arquitectura obtenida puede apreciarse en la sección 17 Anexos.

10.2. Herramientas

Para llevar a cabo la aplicación se utilizaron diversas herramientas de desarrollo:

- Sublime Text
- Ionic
- Además se utilizó la consola de Ubuntu para administrar y probar:
- Base de datos MongoDB
- Conexión y transmisión de mensajes a través de WebSocket
- Cifrado y descifrado de los mensaje

10.3. Lenguajes y Frameworks

El backend fue desarrollado en Ruby mediante el framework Rails. Siendo este lenguaje el que maneja toda la interacción con la base de datos, transmisión y administración de mensajes a través de la librería Action Cable e interacción con el frontend.

El frontend fue desarrollado en JavaScript con HTML y CSS mediante el framework Bootstrap para funcionar en entorno Web.

10.4. Seguridad y Cifrado

Para la seguridad de la aplicación se tuvo como objetivos que la misma esté cifrada punta a punta, es decir, que solo los usuarios que participan de la conversación puedan leer los mensajes y que toda la información almacenada se encuentre cifrada.

En este sentido se tomó como punto de partida los principios del patrón “Secure by Design”, que pone en foco a la seguridad como prioridad número uno en cada etapa del desarrollo y diseño de la aplicación.

Para ello se implementó:

- Los mensajes enviados son cifrados antes de ser enviados mediante el esquema AES de 256 bits, del cual se destacan los usos y las empresas que lo usan en el Marco Teórico la sección 7.2 “Privacidad y Seguridad”, subsección 7.2.4. Que el mensaje viaje cifrado desde su origen y no como texto plano reduce el riesgo de ataques como, por ejemplo, “Man-in-the-middle” ya que en caso de lograr interceptar el mensaje se requiere de la clave para poder leerlo.

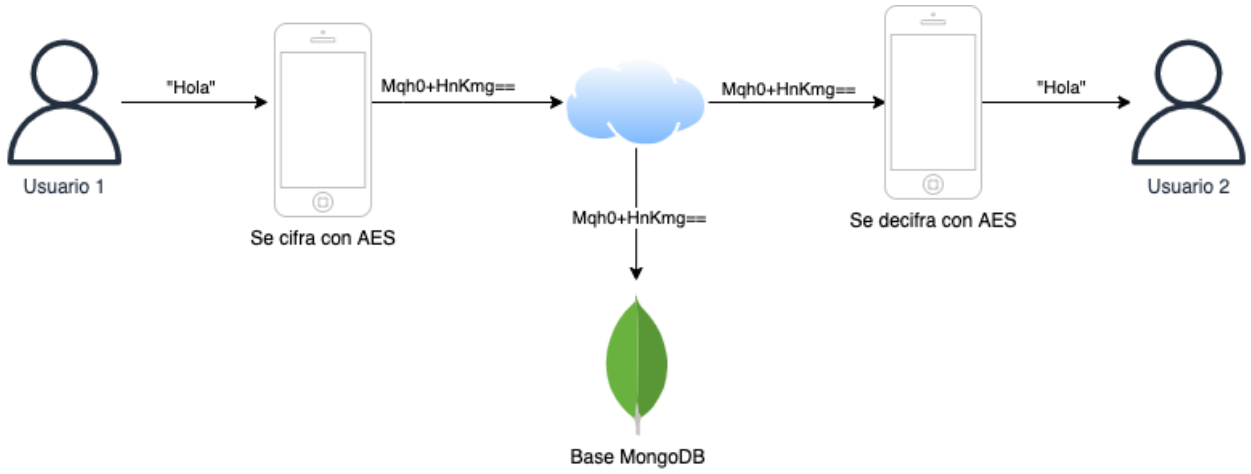


Imagen 32: Cifrado end-to-end

- Todos los mensajes de las conversaciones se almacenan cifrados y nunca como texto plano, como se ilustra en la siguiente captura de mensajes guardados en MongoDB, siendo el campo ¨texto el mensaje cifrado:

- El registro no requiere ninguna información personal de parte de los mismos.
- Las claves generadas para la transmisión de mensajes son cifradas por una clave privada mediante AES de 256 bits asociada a la conversación antes de ser almacenadas. Esta clave nunca es incluida en la transmisión de mensajes y es propia y única de cada conversación.
- Cuando la conversación se elimina, las claves son eliminadas también.

11. Diseño e Interfaz de Usuario

Para incorporar un logo al diseño de la aplicación, mediante Adobe Photoshop, se generaron los siguientes diseños:

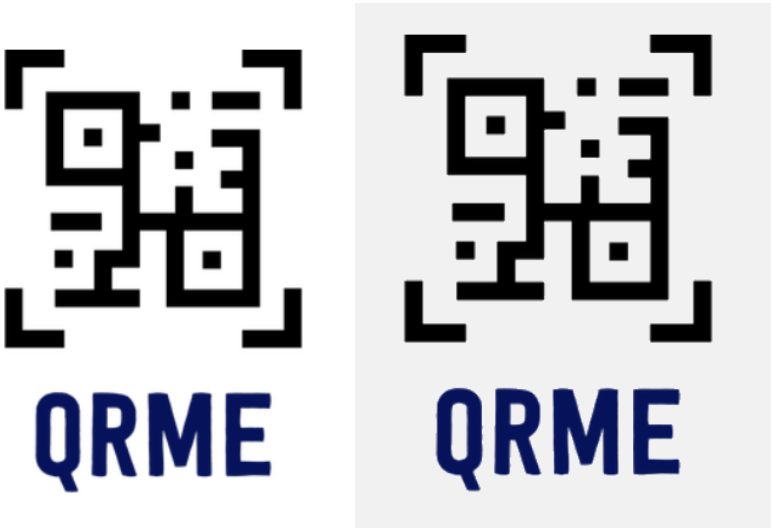


Imagen 33: Logo QRMe

Login de usuario

Imagen 34: Log In QRMe

Registro de usuario

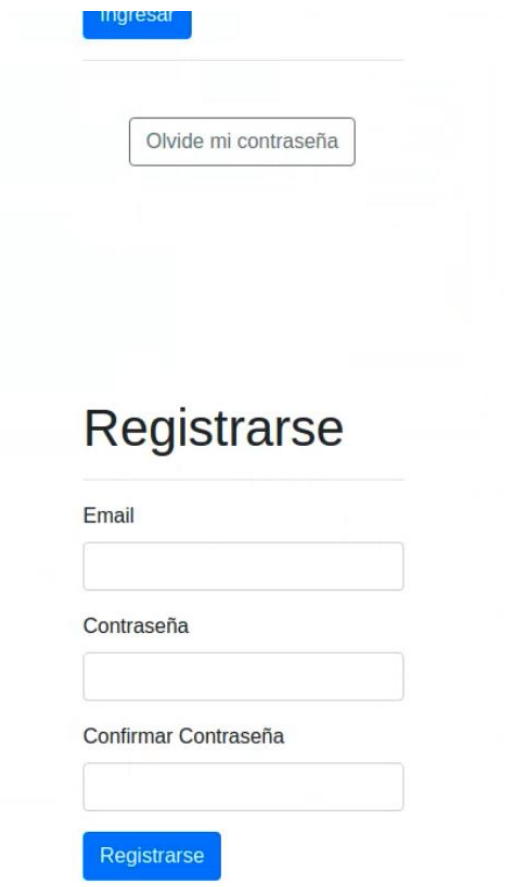


Imagen 35: Registro QRMe

Landing de Usuario Registrado

Imagen 36: Landing QRMe Registrado

Editar datos de Usuario

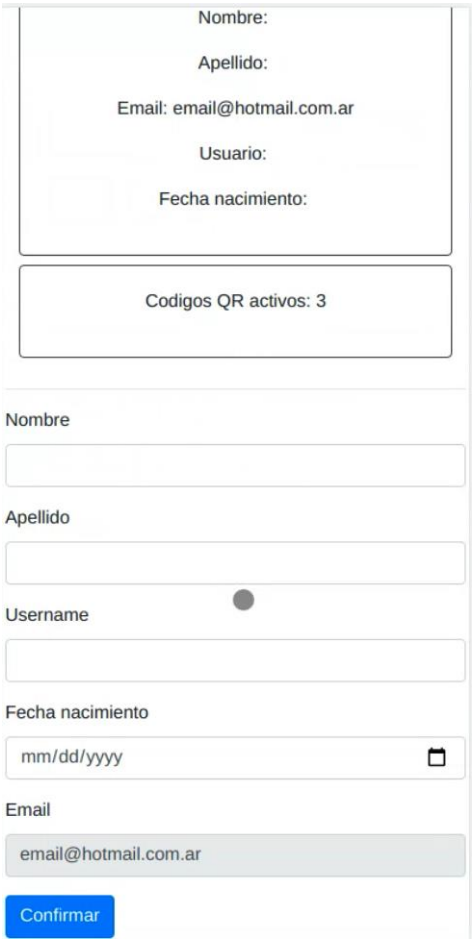


Imagen 37: Editar Usuario QRMe

Crear código QR Parte 1

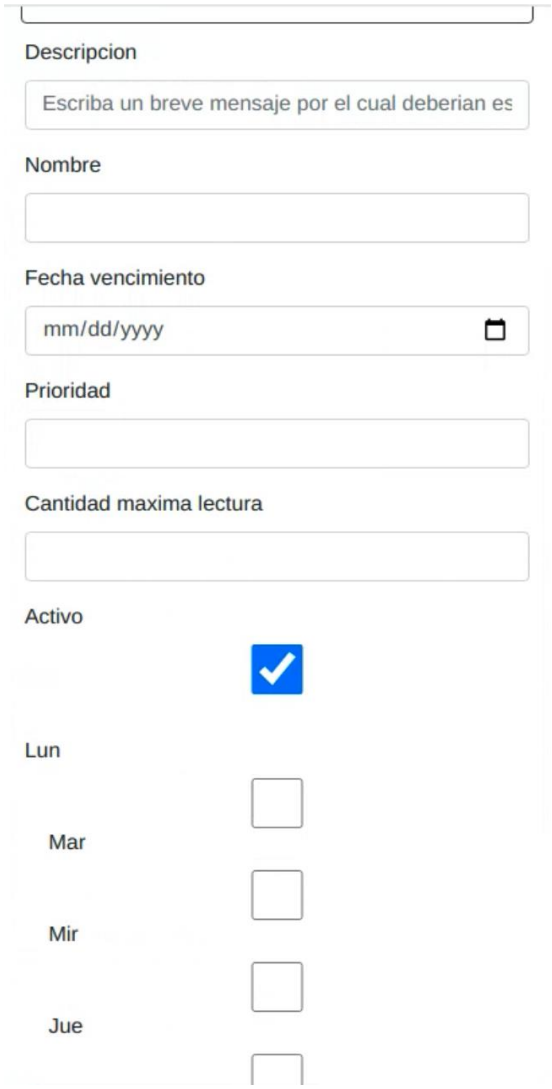


Imagen 38: Crear QR Parte 1

Crear código QR Parte 2

Imagen 39: Crear QR Parte 2

Código QR Exportado en PDF

Imagen 40: Exportar QR

Chat Usuario que escanea QR

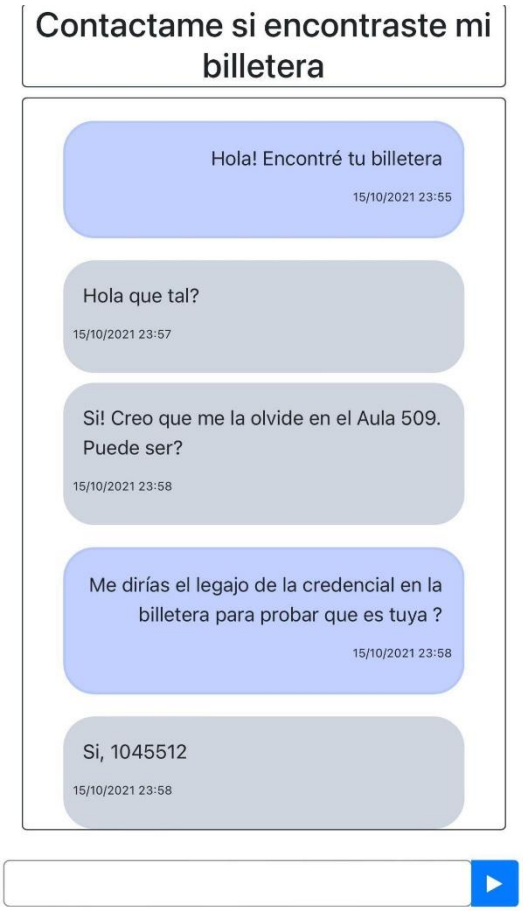


Imagen 41: Chat Usuario que escanea

Chat desde la vista del Usuario dueño del QR en entorno Web de escritorio

Imagen 42: Vista Escritorio Chat

12. Organización del Trabajo

12.1. Versionado de Código

Para el guardado y versionado de código se utilizó el sistema de versionado de código abierto llamado Git, mediante la plataforma GitHub para aprovechar la interfaz visual que provee y utilizando la capa gratuita de dicha plataforma.

GitHub incluye servicios como repositorio, depuración, gestión e implementación de proyectos y código. Dichos repositorios pueden ser tanto privados como públicos, y permiten acceso de distintos niveles a múltiples colaboradores.

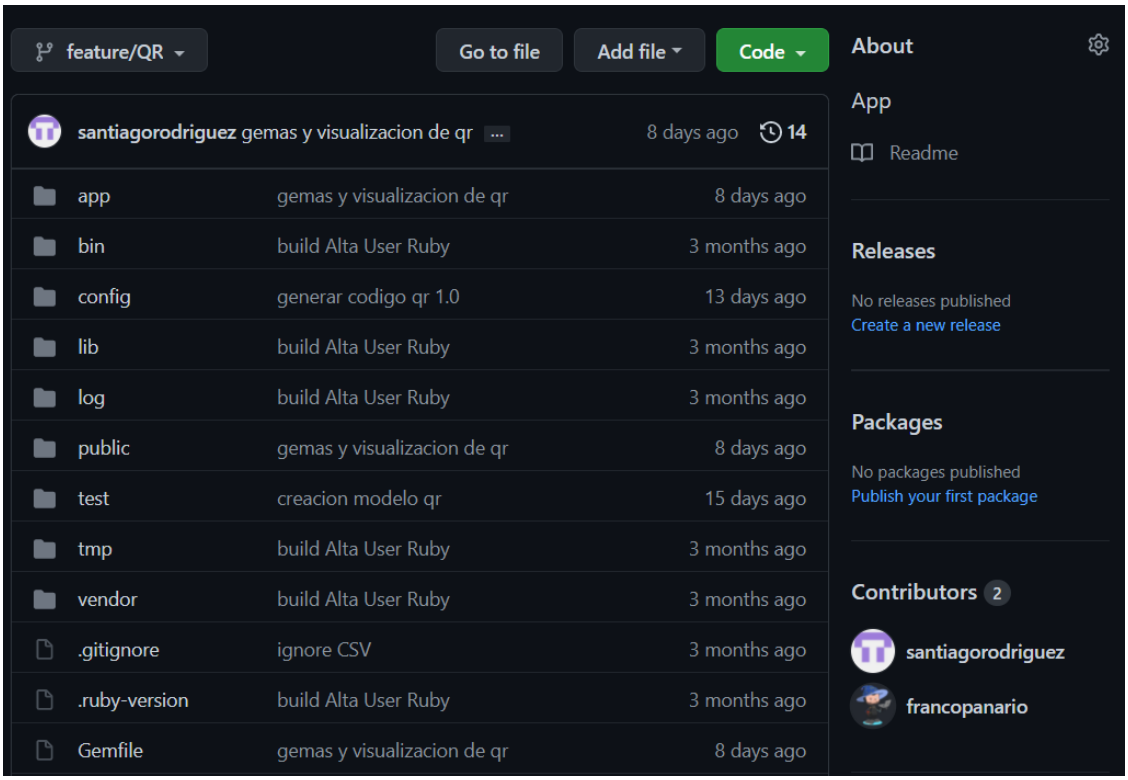


Imagen 43: Repositorio de Código

Al ser un repositorio permite fácilmente compartir proyectos y código con otras personas.

Permite realizar un seguimiento y proporciona un versionado de todos los cambios realizados, además de registrar quien ha realizado dichas modificaciones.

Ofrece integración directa con muchas plataformas de servicios como Amazon Web Services y Google Cloud.

Ofrece toda una sección de ayuda y guías para facilitar la documentación de los procesos y el código. [GITHUB, 2021]

12.2. Organización del trabajo

Dentro de las formas para gestionar el proyecto, se adoptaron herramientas de la metodología Scrum como son los “Sprints”, “Sprints Planning” y “Sprints Retrospectiva”. Se adoptaron dichas herramientas ya que proveen eventos cuyo objetivo es minimizar y optimizar las reuniones. Se definió realizar sprints mensuales, es decir, eventos de un mes donde se definen objetivos y tareas dentro de ese intervalo.

Antes de comenzar el sprint se realiza un Sprint Planning para definir qué objetivos y tareas se pretende abordar. Al final de cada sprint se realiza un Sprint Retrospective para revisar las tareas hechas y analizar las que no se pudieron terminar.

Para poder llevar a cabo la metodología, visualizar las tareas actuales y planificar futuras tareas se utilizó la herramienta Jira de la empresa Atlassian, lo cual permitió armar las historias de usuario y el tablero Kanban.

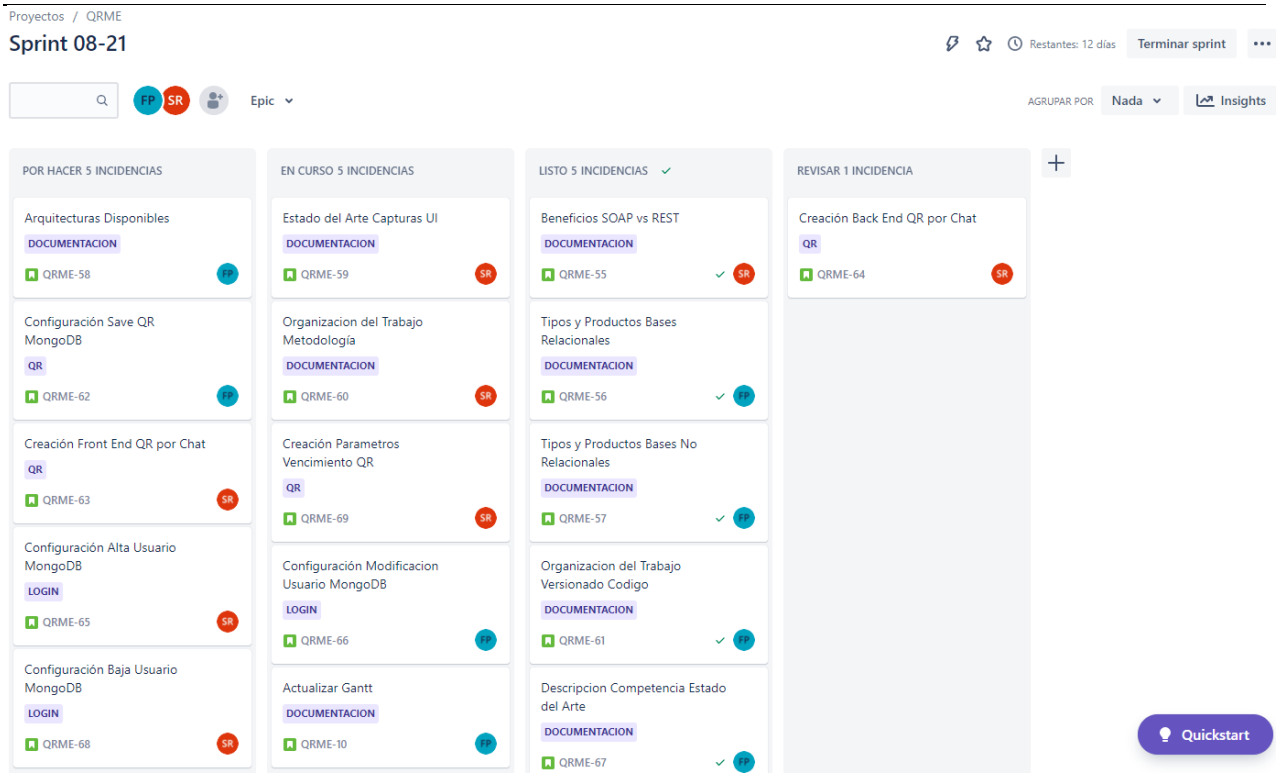


Imagen 44: Manejo Sprint Jira

12.3. Calidad

Para asegurar el buen funcionamiento de la aplicación se utilizó la librería que ofrece el framework Rails llamada RSpec. Esta librería permite generar una serie de pruebas unitarias, las cuales ejecutan las funcionalidades de la aplicación validando su ejecución, tiempos y valores de entrada y salida. A continuación, se listan las funcionalidades probadas incluyendo los módulos que fueron ejecutados por las pruebas unitarias junto con las capturas de imagen de resultado de las pruebas.

Módulo QR

- Creación de QR con atributos obligatorios.
- Ver código QR.

- Modificar características del código QR.
- Eliminar código QR.
- Exportar código QR.
- Generar código QR.

```
1) QR es valido con atributos obligatorios: 'nombre, user y mensaje_motivo'
.2) Ver QR
.3) Modificar QR
.4) Eliminar QR
.5) Exportar QR
.6) Generar QR
.

Finished in 1.73 seconds (files took 0.54116 seconds to load)
6 examples, 0 failures
```

Imagen 45: Test módulo QR

Módulo Mensaje

- Envío de mensaje validando que posea todos los datos asociados necesarios.

```
1) Mensaje es valido con atributos obligatorios: 'fecha, texto y conversacion'
.

Finished in 0.01286 seconds (files took 0.54546 seconds to load)
1 example, 0 failures
```

Imagen 46: Test módulo Mensaje

Módulo Conversación

- Creación de conversación con atributos válidos.
- Ver conversación.
- Modificar el título de la conversación.
- Eliminar Conversación.

```
1) Conversacion es valido con atributos obligatorios: 'fecha, user y qr'
.2) Ver conversacion
.3) Modificar conversacion
.4) Eliminar conversacion
.

Finished in 0.94722 seconds (files took 0.54868 seconds to load)
4 examples, 0 failures
```

Imagen 47: Test módulo Conversación

13. Casos de Uso

Para especificar las funcionalidades de la aplicación y analizarlo desde la perspectiva del usuario se buscó una metodología más referenciable bibliográficamente, a partir de lo cual se optó por utilizar el modelo de casos de uso propuesto por Julie y Kenneth Kendall en su libro *Analisis y Diseño de Sistemas* (edición 2011).

Un caso de uso es un artefacto que define una secuencia de eventos/acciones de la cual se obtiene un resultado de valor observable. Los casos de uso proporcionan una estructura para expresar requisitos funcionales de sistemas informáticos, entre otros sistemas. Pueden representarse como un diagrama, el formato se puede observar en la sección de Anexo 17.2, o como una especificación de caso de uso en un documento textual.

Un caso de uso, de sistema, es una secuencia de acciones que un sistema lleva a cabo que da lugar a un resultado de valor observable para un actor. [IBM, 2016]

Los elementos que componen a un caso de uso para detallar la funcionalidad y la secuencia de eventos que suceden para llegar al resultado de valor son los siguientes:

- Nombre del caso de uso
- ID del caso de uso
- Actor del caso de uso: quien interactúa con el sistema accionando la secuencia de eventos del caso de uso
- Objetivo del caso de uso
- Precondiciones del caso de uso: estado del sistema previo a que el caso de uso comience
- Postcondiciones del caso de uso estado del sistema posterior a la finalización del caso de uso
- Flujos de eventos: secuencia de acciones que, desde estos se alcanza el objetivo del caso de uso
- Información de flujo de evento: Información adicional que puede proporcionarse acerca del flujo

- Escenarios alternativos: si un flujo tiene más de un resultado al ser accionado, se deben detallar aquí los pasos de cada uno de los distintos resultados.

Se muestran a continuación casos de uso para mostrar la metodología, encontrándose en el anexo 17.2 la lista completa:

Nombre: Usuario puede generar conversación sin iniciar sesión a partir de un código QR.	**ID: CU-11**
Actor(es): Usuario	
Objetivo: Generar una conversación nueva desde un explorador Web a partir del escaneo de un código QR.	
Precondiciones: Existe un código QR válido y disponible asociado a un Usuario registrado de QRMe para escanear.	
Postcondiciones: El Usuario se encuentra en una ventana de un explorador Web con una conversación creada a partir del escaneo del código QR.	
Flujo de eventos	**Información**
1. El usuario escanea con su dispositivo el código QR.	
2. El código QR proporciona un enlace al cual el usuario accede.	
3. El sistema recibe la información del enlace accedido, genera una conversación y la asocia al usuario que generó el QR.	
4. El sistema redirige al usuario que escaneó el código QR a una ventana de conversación donde ya puede comenzar a escribir.	

Nombre: Modificar reglas código QR	**ID: CU-07**
Actor(es): Usuario registrado.	
Objetivo: Definir reglas de uso y vencimiento del código QR.	
Precondiciones: El usuario inició sesión y posee al menos un código QR creado.	
Postcondiciones: El usuario modificó las reglas de uso y vencimiento del código QR.	
Flujo de eventos	**Información**
1. El usuario selecciona el botón “Editar” del código QR.	
2. El sistema despliega el menú de reglas disponibles para modificar.	
3. El usuario define las reglas y selecciona el botón “Confirmar”.	
4. El sistema guarda los cambios sobre el código QR	
5. El sistema despliega una lista de las reglas definidas.	
6. El sistema redirige al usuario a su ventana principal.	
Escenarios alternativos	

14. Presupuesto

14.1. Presupuesto del desarrollo de la aplicación

Para el desarrollo de la aplicación se requirieron los siguientes roles:

- Dos desarrolladores Backend: 105 horas cada uno.
- Un desarrollador frontend: 90 horas.
- Un desarrollador móvil: 60 horas.
- Un diseñador UX: 60 horas.
- Dos testers: 60 horas cada uno.
- Un analista funcional: 60 horas.

Con respecto a los costos se contemplaron los siguientes honorarios en dólares estadounidenses por hora: $15 la hora de un desarrollador Backend y de un desarrollador móvil; $13 la hora de un desarrollador Frontend; $12 la hora de un analista funcional y un tester, y $10 la hora de un diseñador UX.

Tabla 2: Presupuesto de desarrollo

Rol	Horas (h)	Costo por hora (U$D/h)	Total por rol (U$D)
Desarrollador Backend	210	15	3150
Desarrollador Frontend	90	13	1170
Desarrollador Móvil	60	15	900
Diseñador UX	60	10	600
Tester	120	12	1440
Analista Funcional	60	12	720
		Total (U$D)	7980

En la tabla 2 se pueden observar los costos por cada rol, con un total de $7.980 dólares americanos para el desarrollo de la aplicación.

Se debe tener en cuenta que el costo de los diferentes proveedores se ve directamente afectado por la cantidad de usuarios que utilizan la aplicación y el uso que le den a la misma.

Se considera que cada usuario registrado realiza en promedio un mínimo de 30 requests al servidor, teniendo en cuenta un inicio de sesión, edición de sus datos, generación de QR, edición de reglas de QR, exportación de código QR, ver lista de QRs generados, vista de conversaciones, vista de mensajes de conversaciones y respuesta de mensajes de conversación. Además se considera que cada usuario no registrado realiza en promedio un mínimo de 10 requests al servidor teniendo en cuenta un escaneo de código QR para generar la conversación y envío de mensajes.

Cada requisito exige un mínimo de un acceso a la base de datos MongoDB, y cada mensaje nuevo transmitido exige un mínimo de un acceso a la base de datos Redis.

14.2. Presupuesto de mantenimiento de la aplicación

Heroku

Para levantar el servidor de la aplicación se utiliza el proveedor Heroku. Dicho proveedor permite comenzar con una capa gratuita con 512 MB de RAM y 2 núcleos de procesamiento.

Por otro lado, Heroku permite escalar tanto en cantidad de requests en paralelo que puede recibir como en cantidad de memoria RAM y núcleos de procesamiento pasando a una capa denominada ¨Standard 2X¨ elevando su capacidad a 1 GB de RAM y eliminando el límite de núcleos de procesamiento, entre otros beneficios. Este escalamiento eleva su costo a 50 dólares por mes.

Asimismo, Heroku ofrece la instancia y acceso a Redis para el almacenamiento temporal de canales de conversación y mensajes, comenzando con una capa gratuita de 25 MB. Permitiendo elevar dicha capacidad de ser necesario. En el caso de QRMe se contempla elevar la capacidad, al momento de que la capa gratuita sea insuficiente, a 100 MB agregando un costo mensual de 30 dólares americanos.

La capa gratuita puede dar servicio a 300 usuarios en simultáneo aproximadamente. Superada esta capacidad se eleva el costo a 80 dólares cada 1200 usuarios.

Mailchimp Mailing

Proveedor de servicio para el envío de correo electrónico. La cantidad de correos que permite enviar se divide en bloques de 25000 correos electrónicos, costando cada bloque 20 dólares americanos por mes.

Se requiere un envío de correo por cada usuario al momento que se registra. Además para cada mensaje de difusión que haga QRMe necesitará tantos envíos como usuarios registrados existan.

No posee capa gratuita por lo que el costo mínimo para integrar a QRMe es de 20 dólares mensuales, agregando 20 dólares mensuales extra por cada 25000 correos más.

MongoDB Serverless

Este servicio de base de datos nativo de MongoDB está montado sobre la infraestructura de AWS y es auto escalable según la demanda de la aplicación.

MongoDB Serverless posee un costo mensual mínimo de 9 dólares por las primeras cinco millones de lecturas y 38 dólares por el primer millón de operaciones de escritura, dando un costo mensual de 47 dólares americanos. Con este servicio, aproximadamente 25.000 usuarios pueden utilizar la aplicación sin inconvenientes.

Por cada 5 millones de lecturas y cada millón de escrituras extra, se eleva el costo mensual 47 dólares americanos más.

El proyecto cuenta con un costo de desarrollo de $7980 dólares americanos. El costo mensual de proveedores en dólares americanos se ve reflejado en la siguiente tabla:

Tabla 3: Presupuesto de desarrollo

Cantidad de Usuarios	**Heroku**	**Mailchimp**	**MongoDB**	**Costo Total**
Hasta 1200	Gratis	20 dólares	47 dólares	67 dólares
Entre 1201 y 25000	Hasta 1680 dólares	20 dólares	47 dólares	1747 dólares
Más de 25000	1680 dólares por cada 25000 usuarios más	20 dólares por cada 25000 usuarios más	47 dólares por cada 25000 usuarios más	1747 dólares por cada 25000 usuarios más

15. Conclusión

La idea qué impulsó este proyecto final de Ingeniería fue ver un auto estacionado en la calle con las balizas encendidas y sin señales del dueño cerca. Esta fue la problemática inicial que se detectó: la imposibilidad de comunicarse con alguien desconocido para alertar de que, en este caso, el auto se puede llegar a quedar sin batería.

A raíz de esta problemática los diferentes escenarios donde alertar y/o informar a un desconocido de otros eventos comenzaron a surgir, por ejemplo: auto estacionado en un lugar no permitido, siniestro a un auto estacionado, pérdida de billetera, recepción de un paquete a domicilio, entre otros.

QRMe resuelve este problema brindando un canal de comunicación anónimo que se puede generar a través de un código QR que, siguiendo con el ejemplo del auto, podría colocarse en el parabrisas.

Vale recordar que la aplicación vela por el anonimato, privacidad y seguridad de ambas partes.

Además, no es condición necesaria estar registrado para escanear un código QR y notificarle a la persona que lo generó sobre el evento en cuestión.

Comparado con las principales aplicaciones de mensajería del mercado actual, las funcionalidades qué destaca a QRMe son: no exigir qué un usuario esté registrado para poder entablar una conversación con un desconocido, el chat se encuentra cifrado por defecto y qué las conversaciones creadas a partir de un código QR estarán regidas bajo configuraciones que el dueño haya preestablecido, como por ejemplo: fecha de vencimiento, días y horarios disponibles a ser escaneado, cantidad máxima de escaneos permitidos, entre otras.

La idea de la aplicación se vió respaldada por la realización de una encuesta orientada a los usos y costumbres de las personas en relación a las aplicaciones de mensajería actuales y también se buscó saber el grado de interés sobre las cualidades qué aporta QRMe al mercado de las aplicaciones de mensajería, qué pueden resumirse en: anonimato por defecto, privacidad y seguridad de las personas, conversaciones con reglas dictadas por quien quiere ser contactado por

un desconocido y la posibilidad qué una persona con su celular pueda escanear un código y comunicarse con otra sin necesidad de estar registrada.

Para la organización del trabajo, se implementaron herramientas de diversas metodologías que mejor se adaptaron al desarrollo de la aplicación. Para el seguimiento y ejecución de los requerimientos se utilizaron reuniones semanales con tareas a realizar, previamente definidas. Estas tareas, junto con su evolución, se documentaron en un tablero de Jira. Las reuniones estaban segmentadas en 3 etapas: en la primera etapa se definía como se iba a resolver la tarea ya elegida a trabajar en el día y se realizaba una investigación sobre las distintas maneras de cómo podía llegar a resolverse. En la segunda etapa se realizaban las implementaciones de las soluciones encontradas e ideadas en la primera etapa. Por último, en la tercera etapa, se evaluaba la situación del proyecto contemplando lo realizado en la segunda etapa y se definía qué requerimientos realizar para la siguiente reunión.

El desarrollo de la aplicación implicó una etapa de análisis y evaluación sobre las tecnologías para cada sección de esta. Esto incluye protocolos de cifrado, protocolo de canal de envío de mensajes, lenguaje y framework para las funcionalidades de una aplicación web y base de datos a utilizar. Como resultado de la etapa se formó la arquitectura detallada en la subsección 17.1 de la sección 17 Anexos.

La realización del proyecto, en su totalidad, fue enriquecedor desde el punto de vista académico y profesional de cada integrante del grupo. Apoyándose en los conocimientos adquiridos durante los años de formación académica se pudo elaborar un trabajo con un nivel de calidad superior a lo ya realizado a lo largo de la carrera. Y en cuanto a lo profesional fue enriquecedor afrontar un desafío qué se considera similar a la realización de un proyecto, es este caso de software dentro de la vida real, participando en cada una de las etapas del desarrollo.

Por último, queremos listar algunas materias de la carrera qué fueron necesarias para llevar a cabo el proyecto:

- Programación I, II Y III
- Base de datos II
- Aplicaciones Interactivas
- Seguridad Informática
- Aplicaciones Distribuidas
- Integración de Aplicaciones
- Arquitectura de Aplicaciones
- Teleinformática y Redes I y II
- Análisis y diseño orientado a objetos
- Seminario de Integración Profesional I y II
- Evaluación de proyectos informáticos
- Dirección de proyectos informáticos

16. Bibliografía

AINHOA LAFUENTE, *Bases de datos relacionales vs no relacionales.* Consulta realizada en *24/07/2021*.

https://aukera.es/blog/bases-de-datos-relacionales-vs-no-relacionales/

AMAZON WEB SERVICES, *NoSQL.* Consulta realizada en *24/07/2021*.

https://aws.amazon.com/es/nosql/

ATLASSIAN KANBAN, *What is a Kanban Board?.* Consulta realizada en *28/07/2021*.

https://www.atlassian.com/agile/kanban/boards

BENJAMIN ANDERSON, *SQL vs NoSQL.* Consulta realizada en *05/08/2021*.

https://www.ibm.com/cloud/blog/sql-vs-nosql

CAS CREMERS, KATRIEL COHN-GORDON, BENJAMIN DOWLING, LUKE GARRATT, DOUGLAS STABILA, *Singal Protocol Analysis.* Consulta realizada en *20/09/2021*.

https://ieeexplore.ieee.org/document/7961996

CHRIS HAUK, *Encrypted Messaging – What Is It, Why Should You Use It And What Are The Best Apps?* Consulta realizada en *27/08//2021*.

https://pixelprivacy.com/resources/encrypted-messaging/

CODE INSTITUTE, *What is a Framework?* Consulta realizada en *25/08//2021*.

https://codeinstitute.net/blog/what-is-a-framework/

DAVE JOHNSON, *What is telegram?. Consulta* realizada en *25/07/2021*.

https://www.businessinsider.com/what-is-telegram

DAVEY WINDER, *Securing NoSQL applications: Best practices for big data security*. Consulta realizada en *22/09//2021*.

https://www.computerweekly.com/tip/Securing-NoSQL-applications-Best-practises-for-big-data-security

DJANGO, *Django makes it easier to build better web apps more quickly and with less code.* Consulta realizada en *26/08//2021*.

https://www.djangoproject.com/

DZONE *Benefits of using GitHub.* Consulta realizada en *02/08/2021.*
https://dzone.com/articles/benefits-of-using-github

FACEBOOK MESSENGER, *How do I start a secret conversation?* Consulta realizada en *25/07/2021.*
https://www.facebook.com/help/messenger-app/811527538946901/?cms_platform=iphone-app&helpref=platform_switcher

FERNANDO TABLADO, *Privacidad Digital.* Consulta realizada en *25/07/2021.*
https://protecciondatos-lopd.com/empresas/privacidad-digital/#Significado_y_concepto

GENETEC, *Why is encryption important? .Consulta* realizada en *23/07/2021.*
https://www.genetec.com/blog/cybersecurity/what-is-encryption-and-how-important-is-it

GIT, *Git SCM.* Consulta realizada en *02/08/2021.*
https://git-scm.com/

GITHUB, *GitHub Features.* Consulta realizada en 02*/08/2021.*
https://github.com/features

GOOGLE PLAYSTORE LINE, *Line: FreeCall and Messenger.* Consulta realizada en *16/06/2021.*
https://play.google.com/store/apps/details?id=jp.naver.line.android&hl=en_US

GRAPH EVERYWHERE, *Bases de Datos NoSQL | Qué son, marcas, tipos y ventajas.* Consulta realizada en *25/07/2021.*
https://www.grapheverywhere.com/bases-de-datos-nosql-marcas-tipos-ventajas/

HIREN DHADUK, *Bootstrap vs React?.* Consulta realizada en *26/08//2021.*
https://www.simform.com/blog/bootstrap-vs-react/

IBM, *Definición de casos de uso.* Consulta realizada en *28/09/2021*
https://www.ibm.com/docs/es/elm/6.0.3?topic=requirements-defining-use-cases

IBM, *SQL vs NoSQL.* Consulta realizada en *27/08/2021.*
https://www.ibm.com/cloud/blog/sql-vs-nosql

INDEED EDITORIAL TEAM, *What is a Web Application?.* Consulta realizada en *25/08//2021.*

https://www.indeed.com/career-advice/career-development/what-is-web-application

IVÁN RAMIREZ, *Comparativa a fondo de aplicaciones de mensajería Android.* Consulta realizada en *26/072021.*

https://www.xatakandroid.com/listas/comparativa-a-fondo-aplicaciones-mensajeria-android-1

JACKIE DOVE, *What is WhatsApp?* Consulta realizada en *25/07/2021.*

https://www.digitaltrends.com/mobile/what-is-whatsapp/

JSON ORG, *Introducción a JSON.* Consulta realizada en *14/07/2021.*

https://www.json.org/json-es.html

JULIE KENDALL, KENNETH KENDALL, Análisis y Diseño de Sistemas. Consulta realizada en *28/09/2021*

http://cotana.informatica.edu.bo/downloads/ld-Analisis%20y%20Diseno%20de%20Sistemas_Kendall-8va.pdf

KASPERSKY, *Encryption.* Consulta realizada en *16/06/2021.*

https://latam.kaspersky.com/resource-center/definitions/encryption

KASPERSKY, *What is a QR Code and is it safe?* Consulta realizada en 17/05/2021. https://www.kaspersky.com/resource-center/definitions/what-is-a-qr-code-how-to-scan

KSENIA ERMOSHINA, FRANCESCA MUSIANI y HARRY HALPIN, *End-to-End Encrypted Messaging Protocols: An Overview.* Consulta realizada en *17/09/2021.*

https://www.researchgate.net/publication/306527072_End-to-End_Encrypted_Messaging_Protocols_An_Overview

KUMAR CHANDRAKANT, *Rest vs WebSockets.* Consulta realizada en *04/09/2021.*

https://www.baeldung.com/rest-vs-websockets

LIFEWIRE FACEBOOK, *Facebook Messenger: Everything you need to know.* Consulta realizada en *25/07//2021.*

https://www.lifewire.com/facebook-messenger-4103719

MARTIN FIETKIEWICZ, *WebSockets vs HTTP*. Consulta realizada en *04/09/2021*.

https://ably.com/topic/websockets-vs-http

MATTHEW O'RIORDAN, *Rails Action Cable: The good and the bad for developers.* Consulta realizada en *27/08//2021*.

https://ably.com/blog/rails-actioncable-the-good-and-the-bad

MEAÑOS FERNANDO, *La cantidad de comercios que aceptan pagos con tarjetas se quintuplicó en los últimos tres años.* Consulta realizada en *22/04//2021*.

https://www.infobae.com/economia/2021/04/26/la-cantidad-de-comercios-que-aceptan-pagos-con-tarjetas-se-quintuplico-en-los-ultimos-tres-anos/

MICHAEL COBB, CORINNE BERNSTEIN, *What is the Advanced Encryption Standard?*. Consulta realizada en *20/09/2021*.

https://searchsecurity.techtarget.com/definition/Advanced-Encryption-Standard

MICHAEL COBB, *Comparing relational database security and NoSQL security*. Consulta realizada en *22/09//2021*.

https://searchsecurity.techtarget.com/answer/Comparing-relational-database-security-and-NoSQL-security

MICROSOFT, *Create Check Constraints.* Consulta realizada en *16/07/2021*.

https://docs.microsoft.com/en-us/sql/relational-databases/tables/create-check-constraints?view=sql-server-ver15

MONGODB, *Advantages of NoSQL*. Consulta realizada en *24/07/2021*.

https://www.mongodb.com/nosql-explained/advantages

MONGODB, *Types of NoSQL Databases* Consulta realizada en *24/07/2021*.

https://www.mongodb.com/scale/types-of-nosql-databases

MOZILLA, *Working with JSON*. Consulta realizada en *14/07/2021*.

https://developer.mozilla.org/en-US/docs/Learn/JavaScript/Objects/JSON

PETE HUNT, *Why React?* Consulta realizada en *26/08//2021*.

https://es.reactjs.org/blog/2013/06/05/why-react.html

PETER KAYERE, *Understanding Socket.io* .Consulta realizada en *27/08//2021.*
https://www.section.io/engineering-education/understanding-socket/

PORT SWIGGER, *What are WebSockets?* Consulta realizada en *10/09/2021.*
https://portswigger.net/web-security/websockets/what-are-websockets

PRATHA AVASHIA, *Ruby on Rails vs Django.* Consulta realizada en *27/08//2021.*
https://www.solutelabs.com/blog/ruby-on-rails-vs-django

PRIVACY INTERNATIONAL ORGANIZATION, *Privacy and why it matters.* Consulta realizada en *26/06/2021.*
https://www.privacyinternational.org/

SCRUM ORG, *What is Scrum?* Consulta realizada en *28/07/2021.*
https://www.scrum.org/

SHUBHI MALL, *Advantages of QR Code.* Consulta realizada en *17/05/2021.* https://scanova.io/blog/blog/2016/02/16/advantages-of-qr-code/

SIMILARWEB MESSAGING APPS, *The most popular messaging apps.* Consulta realizada en *27/07/2021.*
https://www.similarweb.com/corp/blog/research/market-research/worldwide-messaging-apps/

SSL2BUY, *Symmetric vs Asymmetric encryption.* Consulta realizada en *20/09/2021.*
https://www.ssl2buy.com/wiki/symmetric-vs-asymmetric-encryption-what-are-differences

STATISTA, *Most popular global mobile messenger apps.* Consulta realizada en *26/07/2021.*
https://www.statista.com/statistics/258749/most-popular-global-mobile-messenger-apps/

STEPHANE CASTELLANI, *Types of APIs.* Consulta realizada en *16/07/2021.*
https://blog.axway.com/amplify-products/api-management/different-types-apis

SUDHANSHU CHAUHAN, *Anonymity.* Consulta realizada en *25/07/2021.*
https://www.sciencedirect.com/topics/computer-science/anonymity

REDHAT, *What are application programming interfaces?.* Consulta realizada en *16/06/2021*.

https://www.redhat.com/es/topics/api/what-are-application-programming-interfaces

REDHAT, *What's the difference between soap and rest?.* Consulta realizada en *19/07/2021*.

https://www.redhat.com/es/topics/integration/whats-the-difference-between-soap-rest

REDHAT, *What is a REST API?.* Consulta realizada en *18/07/2021*.

https://www.redhat.com/es/topics/api/what-is-a-rest-api

REST vs SOAP, *SOAP vs. REST: A Look at Two Different API Styles.* Consulta realizada en *14/07/2021*.

https://www.upwork.com/resources/soap-vs-rest-a-look-at-two-different-api-styles

ROCK CONTENT, *Bootstrap: que es y para qué sirve?.* Consulta realizada en *25/08//2021*.

https://rockcontent.com/es/blog/bootstrap/

RUBY, *About Ruby.* Consulta realizada en *26/08//2021*.

https://www.ruby-lang.org/en/about/#fn2

RYAN BARONE, *What are libraries in coding?* Consulta realizada en *25/08//2021*.

https://www.idtech.com/blog/what-are-libraries-in-coding

TELEGRAM, *Qué es Telegram y que puedo hacer aquí?* Consulta realizada en *25/07/2021*.

https://telegram.org/faq#p-que-es-telegram-que-puedo-hacer-aqui

VIVIAN MCCALL, *What is WeChat?* Consulta realizada en *25/07/2021*.

https://www.businessinsider.com/what-is-wechat

WHATSAPP SECURITY, *About WhatsApp Security.* Consulta realizada en *25/07/2021*.

https://www.whatsapp.com/security/

WEBSOCKET SECURITY, *WebSocket Security.* Consulta realizada en *10/09/2021*.

https://devcenter.heroku.com/articles/websocket-security

WECHAT DESCRIPTION, *WeChat Help Center.* Consulta realizada en *25/07/2021*.

https://help.wechat.com/cgi-bin/micromsg-bin/oshelpcenter?opcode=2&lang=en&plat=android&pid=1003364&id=120813euejvf141023eumfzy&Channel=WeChatOfficialWebsite

WORLD WIDE WEB CONSORTIUM, *Simple Object Access Protocol.* Consulta realizada en *19/07/2021*.

https://www.w3.org/TR/soap11/#_Toc478383486

17. Anexos

17.1. Diagrama de Arquitectura

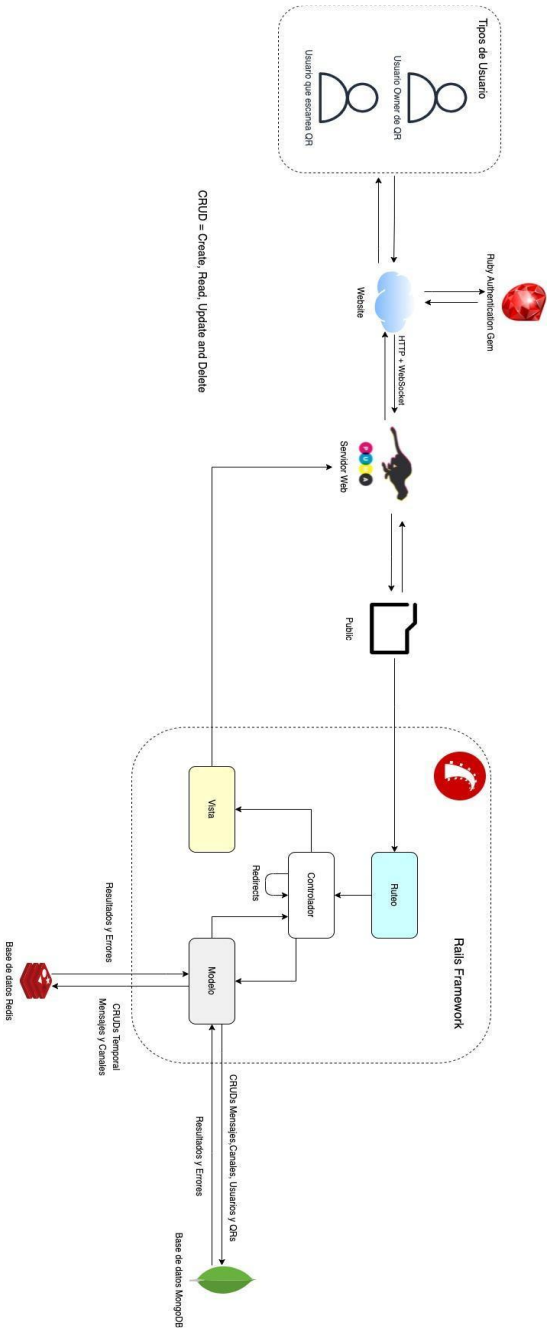


Imagen 48: Diagrama de Arquitectura

17.2. Casos de Uso

Nombre: Registro usuario	**ID: CU-01**
Actor(es): Usuario.	
Objetivo: Permitir que un usuario pueda crear una cuenta ingresando nombre de usuario, email y una clave.	
Precondiciones: No debe tener una cuenta creada con su email y nombre de usuario ingresado.	
Postcondiciones: El usuario obtiene su usuario y la capacidad de ingresar a la aplicación.	
Flujo de eventos	**Información**
1. El usuario accede a la aplicación y en la sección “Registrarse” de la pantalla ingresa los datos solicitados.	
2. El usuario presiona el botón registrarse.	
3. El usuario es dado de alta.	
Escenarios alternativos	
3.1 El usuario no es dado de alta porque su nombre de usuario ingresado ya existe.	
3.2 Repite paso 1.	

Nombre: Login usuario	**ID: CU-02**
Actor(es): Usuario registrado.	
Objetivo: Permitir que el usuario pueda ingresar a la aplicación ingresando email y contraseña.	
Precondiciones: El usuario debe tener una cuenta creada y activa.	
Postcondiciones: El usuario ingresó exitosamente a la aplicación.	
Flujo de eventos	**Información**
1. El usuario ingresa a la aplicación, donde en la pantalla de login selecciona e ingresa email y contraseña.	
2. El usuario selecciona el botón “Iniciar Sesión”.	
3. El sistema muestra la pantalla principal de la aplicación.	
Escenarios alternativos	

Nombre: Cambiar contraseña	**ID: CU-03**
Actor(es): Usuario registrado.	
Objetivo: Permitir que un usuario modifique su contraseña validando su cuenta con su email.	
Precondiciones: El usuario inició sesión.	
Postcondiciones: El usuario modificó su contraseña.	
Flujo de eventos	**Información**
1. El usuario selecciona “Modificar Contraseña” en la sección “Cuenta” del menú arriba a la izquierda.	
2. El sistema muestra el formulario para completar con la nueva contraseña.	
3. El usuario ingresa su nueva contraseña.	
4. El usuario guarda la información ingresada.	
5. El sistema guarda los datos y redirige a la vista de sus datos de cuenta.	
Escenarios alternativos	

Nombre: Recuperar contraseña	**ID: CU-04**
Actor(es): Usuario registrado.	
Objetivo: Permitir que un usuario recupere su contraseña validando dicha cuenta con su email.	
Precondiciones: El usuario debe haber olvidado su contraseña.	
Postcondiciones: El usuario recuperó su cuenta con una nueva contraseña creada por él.	
Flujo de eventos	**Información**
1. El usuario selecciona la opción “Recuperar contraseña”.	
2. El sistema redirige al usuario a un formulario donde le solicita ingresar el email.	
3. El usuario ingresa el email con el qué se registró y selecciona el botón “Confirmar”.	
4. El sistema envía un email qué contiene el link para ingresar una nueva contraseña y así recuperar la cuenta.	
5. El usuario selecciona el link recibido.	
6. El sistema redirige al usuario a una ventana con un formulario para ingresar la contraseña nueva.	
7. El usuario ingresa la contraseña nueva y selecciona el botón “Confirmar”	

6. El sistema reactiva la cuenta y redirige al usuario a la ventana principal de la cuenta recuperada.	
Escenarios alternativos	

Nombre: Modificar datos de la cuenta.	**ID: CU-05**
Actor(es): Usuario registrado.	
Objetivo: Permitir que el usuario modifique los datos asociados a su cuenta.	
Precondiciones: El usuario inició sesión.	
Postcondiciones: El usuario modificó exitosamente sus datos	
Flujo de eventos	**Información**
1. El usuario selecciona “Modificar Datos Personales” en la sección “Cuenta” del menú arriba a la izquierda.	
2. El sistema redirige al usuario a un formulario donde le solicita ingresar su contraseña.	
3. El usuario ingresa la contraseña y selecciona el botón “Confirmar”.	
4. El sistema despliega los datos asociados a la cuenta del usuario.	
5. El usuario modifica los datos que desea y presiona el botón “Modificar”.	
6. El sistema guarda los cambios realizados y redirige al usuario a su ventana principal.	
Escenarios alternativos	

Nombre: Eliminar cuenta	**ID: CU-06**
Actor(es): Usuario registrado.	
Objetivo: Permitir al usuario eliminar su cuenta. Dicha eliminación será una baja lógica.	
Precondiciones: El usuario inició sesión.	
Postcondiciones: Eliminación lógica de la cuenta.	
Flujo de eventos	**Información**
1. El usuario selecciona la opción “Eliminar cuenta” en la sección cuenta del menú arriba a la izquierda.	
2. El sistema redirige al usuario a un formulario donde le solicita ingresar su contraseña.	
3. El usuario ingresa la contraseña y selecciona el botón “Confirmar”.	
4. El sistema despliega una solicitud de confirmación para eliminar la cuenta.	
5. El usuario selecciona el botón “Confirmar”.	
6. El sistema realiza una baja lógica de la cuenta.	
7. El sistema despliega el mensaje “Su cuenta se ha eliminado correctamente”.	
Escenarios alternativos	

Nombre: Modificar reglas código QR	**ID: CU-07**
Actor(es): Usuario registrado.	
Objetivo: Definir reglas de uso y vencimiento del código QR.	
Precondiciones: El usuario inició sesión y posee al menos un código QR creado.	
Postcondiciones: El usuario modificó las reglas de uso y vencimiento del código QR.	
Flujo de eventos	**Información**
1. El usuario selecciona el botón “Editar” del código QR.	
2. El sistema despliega el menú de reglas disponibles para modificar.	
3. El usuario define las reglas y selecciona el botón “Confirmar”.	
4. El sistema guarda los cambios sobre el código QR	
5. El sistema despliega una lista de las reglas definidas.	
6. El sistema redirige al usuario a su ventana principal.	
Escenarios alternativos	

Nombre: Eliminar código QR	**ID: CU-08**
Actor(es): Usuario registrado.	
Objetivo: Eliminar un código QR.	
Precondiciones: El usuario inició sesión y posee al menos un código QR creado.	
Postcondiciones: Se eliminó el código QR.	
Flujo de eventos	**Información**
1. El usuario selecciona el botón “Eliminar” del código QR deseado.	
2. El sistema despliega una advertencia “Si elimina el código QR se eliminaran todas las conversaciones asociadas a él”.	
3. El sistema solicita confirmación de eliminación.	
4. El usuario selecciona el botón “Confirmar”.	
5. El sistema elimina el código QR y todas las conversaciones asociadas a dicho código.	
6. El sistema redirige al usuario a la ventana principal.	
Escenarios alternativos	

Nombre: Generar código QR	**ID: CU-09**
Actor(es): Usuario registrado.	
Objetivo: Crear código QR junto con sus reglas.	
Precondiciones: El usuario inició sesión.	
Postcondiciones: El usuario generó un código QR con reglas definidas	
Flujo de eventos	**Información**
1. El usuario selecciona el botón “Crear código QR”	
2. El sistema despliega un menú con las reglas disponibles para definir.	
3. El usuario define las reglas y selecciona el botón “Confirmar”.	
4. El sistema genera el código QR con las reglas definidas.	
5. El sistema redirige al usuario a la ventana del código QR creado.	
Escenarios alternativos	

Nombre: Exportar código QR	**ID: CU-10**
Actor(es): Usuario registrado.	
Objetivo: Exportar el código QR en formato PDF	
Precondiciones: El usuario inició sesión y posee al menos un código.	
Postcondiciones: El usuario recibe el código QR deseado en formato PDF.	
Flujo de eventos	**Información**
1. El usuario selecciona el botón “Exportar” en el menú del código QR deseado.	
2. El sistema genera un archivo PDF conteniendo el código QR seleccionado.	
3. El sistema despliega al usuario el archivo generado.	
Escenarios alternativos	

Nombre: Usuario puede generar conversación sin iniciar sesión a partir de un código QR.	**ID: CU-11**
Actor(es): Usuario	
Objetivo: Generar una conversación nueva desde un explorador Web a partir del escaneo de un código QR.	
Precondiciones: Existe un código QR válido y disponible asociado a un Usuario registrado de QRMe para escanear.	
Postcondiciones: El Usuario se encuentra en una ventana de un explorador Web con una conversación creada a partir del escaneo del código QR.	
Flujo de eventos	**Información**
1. El usuario escanea con su dispositivo el código QR.	
2. El código QR proporciona un enlace al cual el usuario accede.	
3. El sistema recibe la información del enlace accedido, genera una conversación y la asocia al usuario que generó el QR.	
4. El sistema redirige al usuario que escaneó el código QR a una ventana de conversación donde ya puede comenzar a escribir.	
Escenarios alternativos	

Nombre: Ver mensajes asociados a una conversación.	**ID: CU-12**
Actor(es): Usuario registrado.	
Objetivo: El usuario puede ver los mensajes asociados a una conversación.	
Precondiciones: El usuario inició sesión y posee al menos un código QR creado con una conversación generada.	
Postcondiciones: El usuario ve los mensajes dentro de una conversación.	
Flujo de eventos	**Información**
1. El usuario selecciona el botón “Conversaciones” del menú de opciones.	
2. El sistema despliega la lista de conversaciones creadas.	
3. El usuario selecciona una conversación.	
4. El sistema recupera todos los mensajes asociados a la conversación seleccionada y redirige al usuario a la ventana de conversación donde los muestra.	
Escenarios alternativos	

Nombre: Listar conversaciones asociadas a un código QR	**ID: CU-13**
Actor(es): Usuario registrado.	
Objetivo: Mostrar al usuario todas las conversaciones generadas y asociadas a un código QR específico.	
Precondiciones: El usuario inició sesión y posee al menos un código QR creado.	
Postcondiciones: El usuario ve en pantalla todas las conversaciones asociadas a un código QR.	
Flujo de eventos	**Información**
1. El usuario selecciona el botón “Códigos QR” del menú de opciones.	
2. El sistema despliega una lista de los códigos QR generados por dicho usuario.	

3. El usuario selecciona uno de los códigos QR mostrados en la lista.	
4. El sistema recupera todas las conversaciones asociadas al código seleccionado y redirige al usuario a una ventana donde los muestra.	
Escenarios alternativos	

Nombre: Agregar respuesta automática a conversaciones	**ID: CU-14**
Actor(es): Usuario registrado.	
Objetivo: El usuario pueda definir una respuesta automática en las conversaciones asociadas a un código QR	
Precondiciones: El usuario inició sesión y posee al menos un código QR creado.	
Postcondiciones: Las conversaciones asociadas a un código QR poseen una respuesta automática al ser generadas.	
Flujo de eventos	**Información**
1. El usuario selecciona el botón “Editar” del código QR.	
2. El sistema despliega el menú de reglas disponibles para modificar.	
3. El usuario selecciona la opción “Agregar respuesta automática”.	
4. El sistema despliega un cuadro de texto donde el usuario puede agregar la respuesta automática.	
5. El usuario escribe la respuesta y selecciona el botón “Confirmar”.	

6. El sistema guarda la respuesta y la asocia al código QR.	
7. El sistema muestra un mensaje de confirmación y redirige al usuario a la ventana que lista los códigos QR.	
Escenarios alternativos	

Nombre: Eliminar Conversación	**ID: CU-15**
Actor(es): Usuario registrado.	
Objetivo: El usuario elimina una conversación	
Precondiciones: El usuario inició sesión y posee al menos un código QR creado con al menos una conversación generada.	
Postcondiciones: Se eliminó la conversación.	
Flujo de eventos	**Información**
1. El usuario selecciona el botón “Conversaciones” del menú de opciones.	
2. El sistema despliega la lista de conversaciones creadas.	
3. El usuario selecciona la opción “Borrar” de la conversación.	
4. El sistema despliega una advertencia para confirmar el borrado de la conversación.	
5. El usuario selecciona la opción “Confirmar”.	
6. El sistema elimina la conversación y todos los mensajes asociados a ella.	

7. El sistema redirige al usuario al menú de conversaciones.	
Escenarios alternativos	

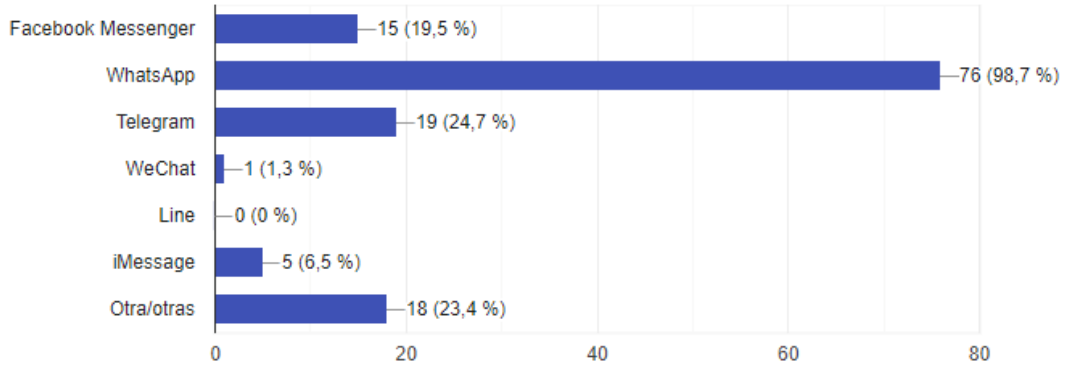
17.3. Encuesta

La encuesta fue llevada a cabo utilizando Google Forms en la cual se realizaron las siguientes preguntas y se obtuvieron los siguientes resultados:

Sección 1

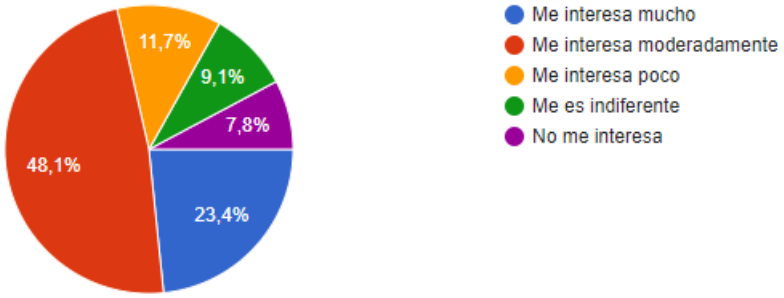
¿Qué aplicación/es de mensajería utilizas?

77 respuestas



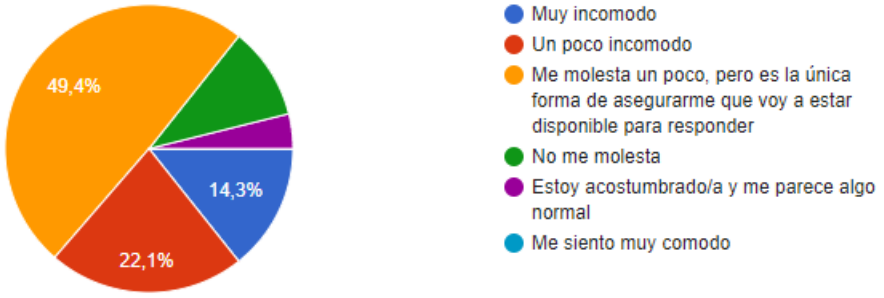
¿Te interesaría que una misma aplicación te permita poder definir múltiples reglas para cada conversación?

77 respuestas



¿Qué tan cómodo/a te sentís compartiendo tu número de teléfono con alguien que debería contactarte por un compromiso especifico y/o esporádico?

77 respuestas



Sección 3

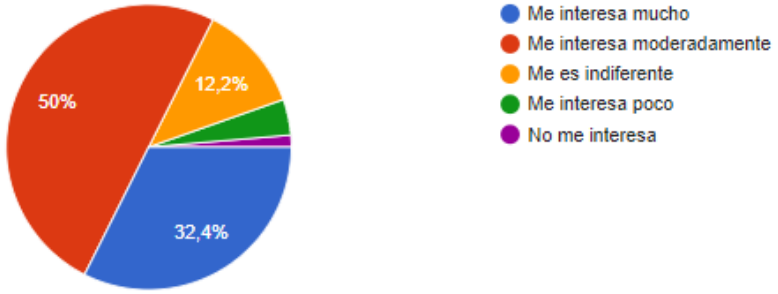
¿Has utilizado alguna vez un código QR?

77 respuestas

Sección 4

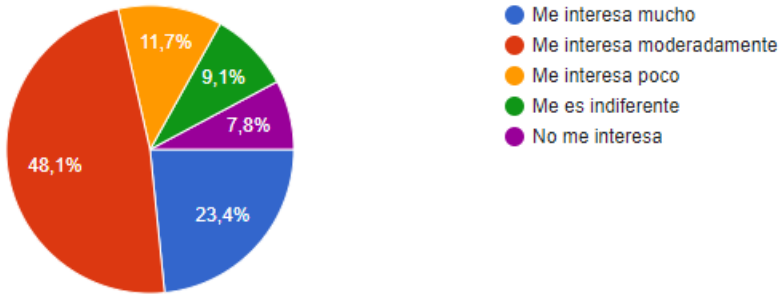
¿Te interesaría poder chatear con otras personas sin la necesidad de compartir tu numero de celular ni ningún otro dato, sino a través de un código QR?

74 respuestas



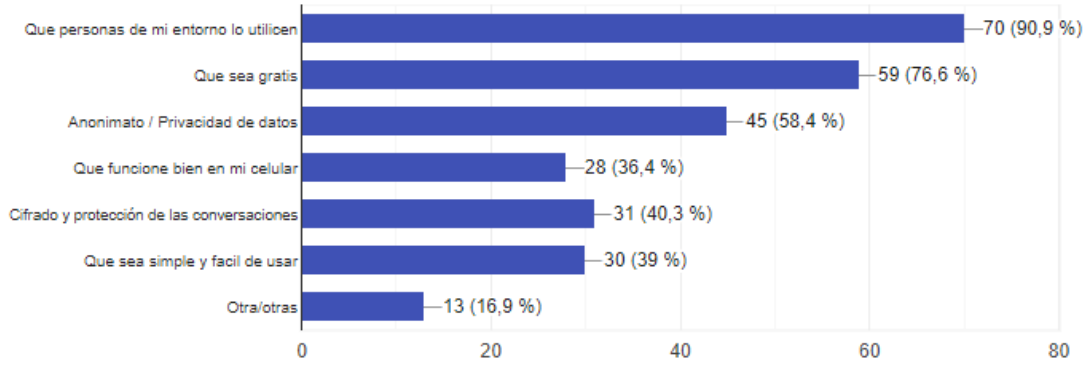
¿Te interesaría que una misma aplicación te permita poder definir múltiples reglas para cada conversación?

77 respuestas



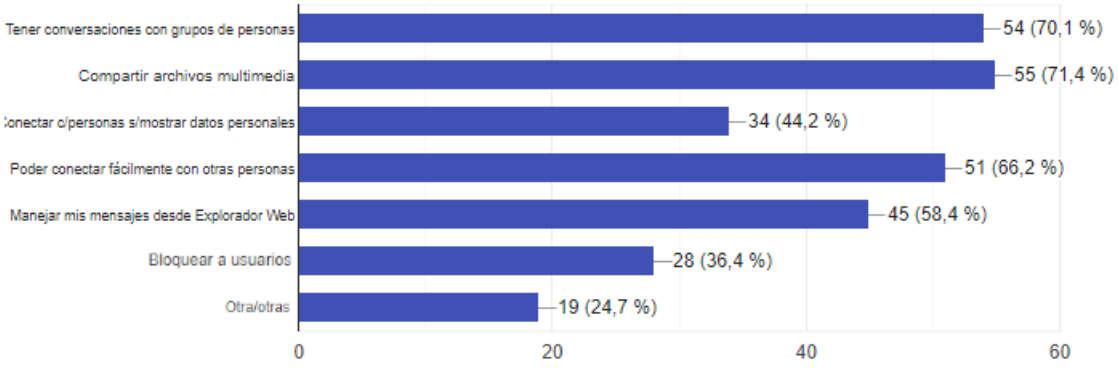
¿Qué cosas son las que tomas en cuenta al momento de elegir que aplicación de mensajería?

77 respuestas



En tu opinión, ¿Qué funcionalidades son las mas importantes en una aplicación de mensajería?

77 respuestas



17.4. Propuesta de Tema

1. Tipo de proyecto

Producto de Software (apunta a obtener un producto de software y/o proceso novedoso)

2. Objetivo

Desarrollar un prototipo de aplicación móvil y web de mensajería instantánea segura que provea la posibilidad de establecer contacto de forma anónima a través de un código QR en Argentina en el año 2021.

3. Alcance

En el primer release el producto estará compuesto por dos interfaces, una móvil resolviéndose a través de una aplicación de celular, y otra web. Dentro de las mismas un usuario podrá generar, exportar y administrar múltiples códigos QR, a través de los cuales un tercero podrá escanearlo y establecer contacto mediante mensajería instantánea. El usuario que genera dichos códigos QR podrá establecer reglas previas sobre el mismo, es decir, validez temporal del código, datos a mostrar una vez establecido el contacto con terceros, escaneos permitidos, respuesta automática y prioridad de notificaciones.

Además, en este release, tanto el usuario dueño de los códigos QR como el usuario qué lo escanea podrán posteriormente agregarse como contactos mutuamente teniendo la capacidad de mantener el anonimato. Estos nuevos contactos agregados estarán disponibles en una agenda de contactos. La conversación inicial quedará guardada en el archivo de conversaciones.

Este producto no incluirá grupos de conversación, agregar etiquetas a lista de contactos, integración con aplicaciones de entrega (por ejemplo: Rappi) o de pago (por ejemplo: MercadoPago), llamadas, videollamadas ni API para ser integrado con aplicaciones de terceros. Sin embargo, son funcionalidades que se agregaran en releases futuros.

4. Descripción

El avance en el mercado de productos como los pagos digitales y, más recientemente debido al COVID-19, el compartir información sin contacto, una mayor cantidad de personas se encuentra familiarizada con dicha tecnología y cómo utilizarla. Este contexto facilita la creación de un producto que permita establecer comunicación entre dos usuarios mediante el uso de dicha tecnología. [Meaños, 2021]

Por otro lado, dentro del mundo de la mensajería instantánea, esta tecnología ya ha dado sus primeros pasos permitiendo que un usuario pueda conectarse con otro, como por ejemplo WhatsApp. Sin embargo, dicho servicio carece de la capacidad de mantener el anonimato de ambas partes ya que al hacer contacto escaneando el código QR, el número de teléfono se hace visible. Además carece de la posibilidad de administrar y personalizar dicho código de tal forma que permita establecer reglas de comunicación previas a que un tercero se contacte con la persona que genera el código. Por último, existe la limitación de poder crear un solo código, removiendo la posibilidad de tener varios con distintas finalidades con sus respectivas categorías y prioridades.

Entendiendo esto, se desarrollará una aplicación que permita establecer una comunicación a través de un canal cifrado y anónimo para ambas partes vía mensajería instantánea. Las características del chat en cuanto a tipos de mensajes permitidos y cantidad de información a brindar a terceros serán configuradas por el usuario qué desea ser contactado. Dentro de la aplicación, el usuario podrá generar múltiples códigos QR, los cuales se les podrán definir múltiples características: prioridad, categoría, información personal a brindar, aceptación de mensajes multimedia, vencimiento, escaneos permitidos, configuración de notificaciones y respuesta automática.

El usuario deberá poder descargarse la aplicación al celular de forma gratuita, dentro de la cual podrá registrarse sin la obligación de brindar un número de teléfono. Los usuarios que escanean el código deberán poder establecer conversaciones con el usuario dueño del código sin la necesidad de descargar la aplicación al celular, es decir, podrán intercambiar mensajes desde una ventana revolviéndose en un explorador web. Tanto como para escanear el QR y conversar con su

creador como para agregar terceros como contactos será condición necesaria que se encuentren registrados en la aplicación.

El usuario que desea ser contactado de forma anónima por cualquier tipo de motivo específico puede colocar el código QR dentro de un objeto que, en caso de ser extraviado, un tercero pueda escanearlo y comunicarse de forma directa, o en su puerta en caso de que esté esperando un paquete y necesiten notificarle la entrega, o en su vehículo en venta para qué posibles compradores puedan contactarlo, o colocar el código en la mochila de colegio de su hijo para ser contactado por cualquier cuestión de inmediato, entre muchos otros motivos que pueden ser relevantes.

5. Aportes

El desarrollo efectivo de la aplicación brindará la posibilidad de que cualquier persona con un teléfono móvil e internet pueda establecer comunicación mediante mensajería instantánea con terceros a través de un medio que por defecto no brinda datos personales de ninguno de los participantes. En este sentido se aumenta la seguridad del usuario en lo referido a la protección de su privacidad. Es decir, si consideramos como datos sensibles de una persona el nombre, apellido y el número de celular para que dos personas intercambien mensajes en WhatsApp, por ejemplo, ya estarían siendo forzados a ser revelados, mientras que utilizando QRMe no será necesario.

Por otro lado, la aplicación proporcionará la capacidad de tener varios códigos QR para poder ser contactado por diferentes motivos, de los cuales el usuario será el único propietario y podrá disponer de su uso y duración como desee.

Asimismo, se eliminará la necesidad de tener instalada la aplicación en el celular para el usuario que escanee el código QR ya que el mismo podrá comunicarse desde una interfaz web de forma directa.

Por último, esta aplicación proporciona al usuario total libertad sobre las conversaciones, dándole la capacidad de configurar y moderarlas una vez iniciadas tras la lectura de un código QR. Es decir, las conversaciones tendrán reglas establecidas por quien haya creado el código. Y

posterior al momento de inicio de una comunicación tanto los usuarios como el intercambio de mensajes pueden quedar guardados en una lista de contactos y de conversaciones respectivamente.

6. Recursos

INSUMO	DESCRIPCIÓN	CANTIDAD	COSTO ESTIMADO ($)
Computadora	Personales	2	0
Ruby on Rails	Lenguaje de programación aplicación BackEnd Ruby 2.3.1 Rails 5.2.3	1	0
MongoDB	Base de datos	1	0
Bootstrap 4.3.1	Framework FrontEnd	1	0
TOTAL, PRESUPUESTO ESTIMADO: $0			

7. Cronograma Tentativo de Actividades

Cronograma tentativo de tareas	Comienzo	Final
Armado y presentación de propuesta de tema	01/04/2021	24/04/2021
Definición de metodología de trabajo	22/04/2021	24/04/2021
Definición de lenguaje y tecnologías de desarrollo	26/04/2021	10/05/2021
Definición de arquitectura de la aplicación	11/05/2021	17/05/2021
Desarrollo y diseño de funcionalidad Login	11/05/2021	27/08/2021
Armado modelo código QR	25/05/2021	02/06/2021
Informe de avance I	05/06/2021	05/06/2021
Presentación Preliminar I	05/06/2021	05/06/2021
Investigación de protocolos de seguridad	07/06/2021	07/07/2021
Ideación de diseño Web	11/06/2021	18/06/2021
Desarrollo Funcionalidad Mensajería	07/06/2021	04/11/2021
Diseño y testeo Experiencia Usuario	01/07/2021	25/07/2021
Informe de avance II	21/08/2021	21/08/2021
Presentación Preliminar II	21/08/2021	21/08/2021
Desarrollo funcionalidad Manejo QR	01/07/2021	27/09/2021
Configuración de alojamiento de la aplicación	15/07/2021	27/09/2021
Ejecución del diseño web responsive	01/08/2021	25/08/2021
Pruebas de estrés de la aplicacion	07/08/2021	25/09/2021
Informe de avance III	16/10/2021	16/10/2021
Presentación Preliminar III	16/10/2021	16/10/2021
Conclusión	17/10/2021	27/10/2021
Presentación final	27/10/2021	27/10/2021
Defensa PFI	11/12/2021	11/12/2021