ESCUELA POLITÉCNICA NACIONAL

ESCUELA DE FORMACIÓN DE TECNÓLOGOS

DESARROLLO DE APLICACIÓN MOVIL DE GEOLOCALIZACION PARA ALARMA Y NOTIFICACION DE PANICO EN PLATAFORMA ANDROID

INGENIERIA DE SOFTWARE

TRABAJO DE INTEGRACIÓN CURRICULAR PRESENTADO COMO REQUISITO PARA LA OBTENCIÓN DEL TÍTULO DE TECNÓLOGO SUPERIOR EN DESARROLLO DE SOFTWARE

CARLOS ANDRE MONTESDEOCA GARCIA

ING. JUAN PABLO ZALDUMBIDE PROAÑO
DIRECTOR

DMQ, febrero 2022

CERTIFICACIONES

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

Carlos André Montesdeoca García

carlos.montesdeoca01@epn.edu.ec

carlos_andre_mg@outlook.com

Certifico que el presente trabajo de integración curricular fue desarrollado por Carlos André Montesdeoca García, bajo mi supervisión.

Ing. Juan Pablo Zaldumbide Proaño

DECLARACIÓN DE AUTORÍA

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

Carlos André Montesdeoca García

DEDICATORIA

El presente trabajo está dedicado a mis familiares como padre, madre, hermanos, quienes estuvieron apoyándome y motivándome para jamás rendirme en este largo y gratificante camino de formación profesional.

CARLOS ANDRE MONTESDEOCA GARCIA

AGRADECIMIENTO

A mi padre que me han inculcado el valor de la perseverancia y poner la frente en alto aun que las cosas parezcan difíciles, para alcanzar los objetivos que me he propuesto y sobre todo que rendirse no es una opción.

A mi madre que tuvo la paciencia y atención asía mi persona apoyándome en cada momento y dificultad que se me atravesaba.

Un especial agradecimiento a los ingenieros que tuve la oportunidad de conocer a lo largo de mi formación académica dentro de esta maravillosa universidad, ya que por su dedicación y pasión en su trabajo lograban inspirar a muchos estudiantes a continuar y no rendirse en la carrera, además de ser grandes amigos dentro y fuera de las actividades académicas.

CARLOS ANDRE MONTESDEOCA GARCIA

ÍNDICE DE CONTENIDO

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

RESUMEN

El presente proyecto surge como una idea de seguridad ciudadana, destinada a la población ecuatoriana, ya que, como consecuencia de la baja económica sufrida en el país por la pandemia, los robos en las calles dieron un notable incremento.

Considerando que los ciudadanos se ven en la obligación y necesidad de movilizarse a diario para sustentar sus hogares, siendo los momentos donde se ven más vulnerables a ser víctimas de un acto delictivo. Por tal motivo existe un miedo en la población ya que en ocasiones los delincuentes pueden robarles sus herramientas necesarias para cumplir con sus actividades laborales como por ejemplo los repartidores.

Por tal motivo se ha creado un aplicativo móvil que permita usar los teléfonos como un medio de alerta en caso de ser necesario, y para dar una ayuda inmediata también se ha incluido una forma de localizar a la víctima.

El proyecto se ha desarrollado ha desarrollado con una metodología de Scrum para mejorar el crecimiento progresivo del desarrollo, y se usó Ionic framework para agilizar la codificación por si compatibilidad con diferentes tecnologías. Y para gestionar el almacenamiento de los datos de este se usó una base de datos NoSQL. Con el uso en conjunto de dichas herramientas se logró concluir el proyecto a través de entregas de funcionalidades pequeñas que se complementaban poco a poco hasta formar las funcionalidades requeridas.

ABSTRACT

This project is an idea of citizen security, aimed at the Ecuadorian population, since, because of the economic downturn suffered in the country by the pandemic, robberies on the streets increased significantly.

Considering that citizens are obliged and need to mobilize every day to support their homes, being the moments when they are most vulnerable to being victims of a criminal act. For this reason, there is fear among the population, as criminals may sometimes steal their tools needed to carry out their work, such as delivery men.

For this reason, a mobile application has been created to use phones as a means of alerting if necessary, and a way to locate the victim has also been included to provide immediate help.

The project has been developed with a Scrum methodology to improve the progressive growth of the development, and Ionic framework was used to streamline the coding for compatibility with different technologies. A NoSQL database was used to manage the storage of this data. With the combined use of these tools, it was possible to complete the project by delivering small functionalities that were gradually supplemented to form the required functionalities.

1 DESCRIPCIÓN DEL COMPONENTE DESARROLLADO

Las tecnologías modernas han permitido mejorar los dispositivos móviles creando un gran catálogo de tipos según si modelo, funciones, gama, entre otras características, lo que provoca que cada vez los precios de estos dispositivos se eleven. provocando un aumento en la economía de los países en el ámbito tecnológico, pero también creando focos de concentración para la delincuencia que se enfoca en el robo para posventa de los dispositivos sustraídos. solo en año 2021 luego de la casi reactivación de las actividades a nivel nacional se registraron 18.221 mil robos a personas, siendo el 34% en la tarde [7].

Actualmente los teléfonos son usados para muchas actividades tanto laborales como personales y de ocio, por lo que cada vez más personas usan entre uno a dos dispositivos dividiéndolos en actividades laborales o de uso personal viéndose reflejado en las estadísticas entre el año 2020 y 2021 presentando una relación de 17.77 millones de pobladores vs 13.82 millones de dispositivos móviles conectados a la red [6]. lo cual representa casi un 75% de la población con acceso a internet a través de un dispositivo móvil.

Además, es importante reconocer que los dispositivos móviles ya no son solo unos teléfonos, sino que dentro de ellos existe una infinidad de funciones creadas y que pueden ser creadas, gracias a la facilidad de desarrollo que ofrece el sistema operativo Android [25]. esta facilidad de desarrollo en esta plataforma junto con la presente situación de seguridad que atraviesa el país motiva a crear nuevos sistemas de ayuda a la población, que pueda trabajar en conjunto a el ECU 911 [11].

Considerando todo lo anterior, se desarrolló una aplicación móvil destinada a las plataformas Android que permita a los usuarios tener un método seguro de seguridad y alerta ciudadana, aprovechando las funcionalidades internas del teléfono, como su función de GPS y aprovechando las librerías eternas que brinden servicios adicionales en la aplicación. de esta forma el usuario tendrá a la mano una primera versión del aplicativo que permita generar notificaciones de alerta a sus contactos de confianza.

1.1 Objetivo general

Desarrollar una aplicación móvil para la plataforma Android, que permita a los usuarios tener un botón de pánico para alerta de ayuda y la geolocalización a otros usuarios cercanos

1.2 Objetivos específicos

- Analizar los requerimientos correspondientes para la implementación del sistema.
- Diseñar la arquitectura de la solución basada en los requerimientos.
- Implementar la plataforma móvil en función del diseño.
- Realizar las pruebas en la plataforma móvil.

1.3 Alcance

La ciudadanía en general se siente cada día más insegura por el aumento de los índices delictivos del país, lo que ocasiona temor en las personas por no saber cómo contrarrestar a dichos actos que van en contra de la integridad personal, hablando de los métodos para frenar los actos delictivos dentro de lo que es el hurto o robo de dispositivos móviles existen algunas alternativas como un ejemplo el bloqueo de servicios para un dispositivo reportado como robado.

Aun que son acciones efectivas cada, cada vez es más difícil la eficiencia de dichos procesos, ya que algunas tecnologías mejoran para mal, facilitando las acciones para contrarrestar esos métodos de seguridad.

Por tales motivos se ha desarrollado un aplicativo móvil llamado **“BOTON DE PANICO”,** para ayudar a los usuarios de dispositivos Android en beneficio a su seguridad, el sistema cuenta con un solo tipo de usuario por lo que las únicas funciones que se presentan son:

- Configuración de perfil, donde estará su nombre y un mensaje personalizado o dejar el mensaje por defecto.
- Crear una pequeña agenda de contactos a quienes estará destinada la alerta de pánico.
- Accionar el botón de notificación de pánico del sistema.

1.4 Marco teórico

Metodología

Como en todo proceso practico de investigación o validación de una hipótesis, se requiere validar las ideas planteadas y verificar su confiabilidad. De tal forma es como una metodología existe, ya que son esa serie de procesos sistémicos que se aplican a lo largo de una investigación o proceso de validación científica, la cual debe ser metódicamente planificada y organizada para lograr validad las ideas planteadas en la hipótesis que se definan al principio de cualquier proceso científico o de investigación.

Existen barrio tipos de metodologías, cada una con sus puntos fuertes y débiles en el momento de su aplicación, pero para eso es importante saber reconocer los parámetros y requisitos que se requieren en el proceso de investigación científica, para poder definir cuál metodología es la más adecuada para la aplicación en un proyecto. Ya que, si bien cada una cumple el objetivo de demostrar las hipótesis planteadas, cada una así mismo se desempeña en campos donde influyen diferentes factores o participantes en el proceso.

Metodología Ágil

Como una rama de las diferentes metodologías de investigación, también se vio la necesidad de crear un procesos sistemático y metódico para el desarrollo de software, todo con el fin de estandarizar el proceso de desarrollo.

El propósito de aplicar una metodología dentro del desarrollo de software es tomar las mejores prácticas de un proceso de investigación y adaptarlos a la creación o ingeniería de software, presentando como diferencia que se debe tomar en cuenta la agilidad y entregas crecientes de un producto en estimaciones de tiempo.

Una metodología ágil también tiene sus diferentes tipos, que su aplicación va variando, dependiendo de los procesos requeridos por un cliente, y los participantes dentro de un proceso de desarrollo.

Back-End

La primera capa de una aplicación web se maneja por la lógica de los procesos y conexiones necesarias para un uso fluido que a la final se presentara al usuario.

El back-end es una capa imperceptible para el usuario de un sistema de software, ya que el desarrollo del mismo está destinado a que la capa superior que interactúa con el usuario

es el que lo gestione, haciendo las peticiones a un servicio. Dentro de las funciones principales de un back-end se pueden encontrar:

- Gestiona las solicitudes que el usuario hace
- Conecta con la base de datos
- Gestiona los métodos de seguridad informática
- Optimiza y gestiona los recursos informáticos para el usuario final.

Front-End

La capa que conecta con el usuario final y el back-end de la aplicación se denomina Front-end, es la capa más grafica que el usuario si puede manipular e interactuar con el, esta capa debe ser tener un estilo grafico adecuado para el giro del negocio donde se esté desempeñando y contar con la facilidad de navegabilidad que un cliente puede necesitar para su uso.

Esta capa también es un puente de comunicación que recibe las peticiones de un usuario a través de las diferentes funciones programadas en el para posteriormente mandarlas al Back-end, pero también tiene sus propios procesos que son más simples pero muy importantes como el enrutamiento entre sus diferentes páginas, transformaciones básicas de datos, envía los métodos que se solicitan a una base de datos (GET, POST, PUT, DELETE).

Aplicación móvil

Las aplicaciones móviles son una distribución más en el desarrollo de software, pero a diferencia de un sistema web lo que una aplicación móvil proporciona es una función aislada y limitada.

De la misma forma que un sistema de software que está diseñado para recibir peticiones de parte de un cliente y consultar a una base de datos, estas también están formadas por capas, como lo son el Back-end y el Front-end. Que en conjunto crear un sistema homogéneo que permite formar una experiencia integra a usuario.

Pero con el avance de las tecnologías y la necesidad humana de integrar los dispositivos móviles en las tareas cotidianas de la industria o actividades diarias, las aplicaciones móviles también han ido evolucionando e integrándose a nuevos campos que van más allá de una codificación únicamente móvil, buscando de esta forma una compatibilidad entre

sistemas operativos e incluso en sistemas de software de otras naturalezas por lo que también existe una distribución más entre una aplicación móvil.

Aplicaciones nativas

Son aplicaciones que están destinadas a un único sistema operativo, todo con el fin de aprovechas todas las ventajas de los servicios que ofrecen los sistemas Android o IOS.

Actualmente las aplicaciones se enfocan más en la creación de sistemas nativos que esto mejora las experiencias porque da una identidad al sistema operativo en el que se trabaja, pero a su vez aumentan los costos de desarrollo y mantenimiento por el hecho de tener que trabajar en una aplicación para dos sistemas totalmente diferentes.

Aplicaciones hibridas

Son aplicaciones que se disfrazan como una aplicación móvil a pesar de ser desarrolladas con tecnologías de sistemas web, aunque es una solución rápida para la creación de nuevos sistemas y su mantenimiento, también es una desventaja puesto que carece de velocidad y capacidad de procesamiento ya que no aprovechan las características de un sistema operativo.

2 METODOLOGÍA

En la ingeniería de software es bastante recomendó el uso de metodologías agiles para su desarrollo, puesto que es una forma de trabajar garantizando la flexibilidad y trabajo en equipo [15], y con ayuda de una metodología Kanban ayuda a agilizar y organizar aún más los procesos ya que las tareas pendientes pueden ser categorizadas en un tablero por orden de importación para el éxito del proyecto [22].

En la metodología Scrum se manejan una serie de procesos como manejo de trabajo en equipo y cronogramas de tareas, todos divididos en diferentes bloques denominados Sprints [16]. De esta forma esta metodología es adecuada para este proyecto ya que con la entrega correspondiente de los Sprints definidos en el manual técnico, se puede evidencian el proceso de desarrollo de una forma incremental y organizada.

2.1 Metodología de Desarrollo

Para la metodología Scrum es importante la estimación de tiempos y el desarrollo incremental del proyecto conforme se dan de baja las actividades realizadas, esto es importante ya que permite tener un avance más notorio y satisfactorio por parte del cliente, de esta forma la relación con el Product Owner se ve beneficiada al momento de querer establecer los procesos de recopilación de requerimientos, diseños y diagramas que permitan dar a comprender la complejidad e importancia del proyecto y sus funciones internas.

Roles

La metodología Scrum está basado en la distinción de Roles donde se incluyen todos los implicados del proyecto, quienes tiene una función importante por lo que su participación es necesaria para esta metodología. En la filosofía de Scrum es importante la integración y participación para una buena planificación de las actividades a realizar en un proyecto y alcanzar el éxito de este.

Product Owner

Puede ser conocido como uno de los dueños del negocio, o implicado principal, por lo cual es quien conoce y proporciona la información necesaria para conocer las necesidades del mercado al que se aplicara el proyecto [17]. En este caso quien actúa como el cliente del producto brindando la información adecuada para crear una aplicación acorde a las sugerencias del usuario son contactos de mi persona con los que he tenido la oportunidad de reunirme presencialmente.

Scrum Master

Es el líder del equipo de desarrollo, está enfocado en trasmitir las ideas del Product Owner al equipo de desarrollo, es importante que el Scrum Master seca congeniar con el cliente para poder integrase a la visión del negocio [17]. El director designado para el proyecto actúa en este rol ya que por su experiencia tanto en actividades de desarrollo y relaciones de negocio, está más que capacitado para ayudar a resolver cualquier duda y alcanzar las metas dispuestas para el éxito del proyecto.

Development Team

es un equipo formado con las habilidades técnicas adecuadas para la ejecución y codificación del proyecto para alcanzar la visión y objetivos de un cliente [17]. El autor del proyecto es quien actúa como el equipo de desarrollo, ya que se tiene las capacidades adecuadas para poder ejecutar el proyecto en el periodo de tiempo establecido.

Recopilación de Requerimientos

Para el desarrollo de software es muy importante establecer las funciones con el Product Owner, ya que de esta forma se establecen los requisitos que presentan importancia en el giro del negocio y que otras opciones pueden ser descartadas. En este caso para recopilar la información se realizó unas reuniones a usuarios Android, dando como resultado confirmando algunas hipótesis con respecto a algunas funcionalidades y aportando con nuevas ideas, esta lista de requerimientos está disponible en el **Anexo II** en la sección de **Requerimientos**.

Casos de uso

Una vez finalizado el proceso de recopilación de requerimientos se lo analizo para poder crear un diagrama de casos de uso que permita demostrar de una forma sencilla u rápida como el usuario interactúa con la aplicación, también algunas acciones internas que se realizan dentro del sistema antes de poder ser enviado a los otros actores participantes, que en este caso no necesariamente corresponde a un usuario del sistema, más bien es un contacto del usuario.

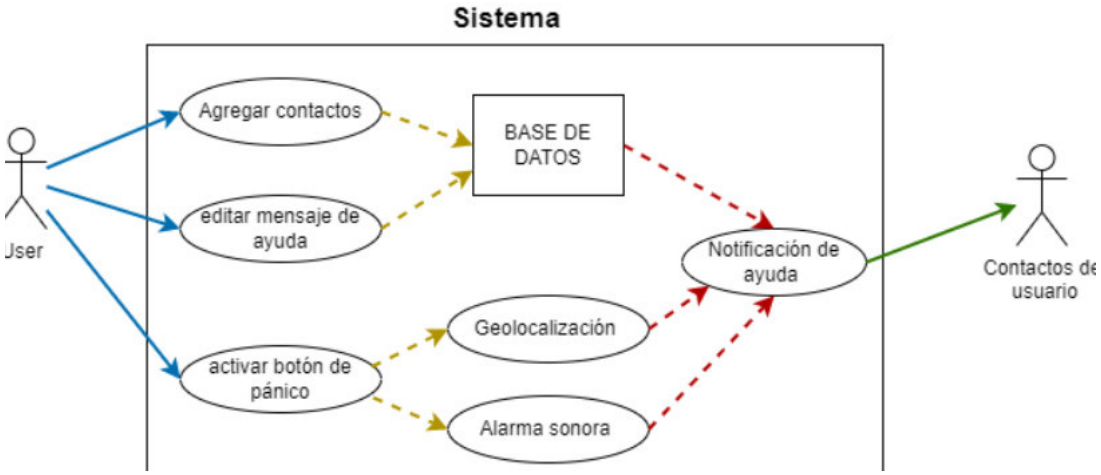


Fig 1. Diagrama de casos de uso

Historias de Usuario

Para la recolección y distinción de las diferentes funcionalidades del sistema se usan unas cartas o tablas donde se incluye una información corta y descriptiva que incluye la funcionalidad y el usuario al que ira destinado a dicha acción, en la **TABLA I.** Se muestra un ejemplo sacado del **Anexo II**, ahí se pueden encontrar el resto de tablas. Con esto se logra clasificar y organizar las diferentes actividades a realizar en el proyecto.

TABLA I: historia de usuario 2 - botón de alarma

HISTORIA DE USUARIO	
Identificador: HU-02	**Usuario:** usuario
Nombre de historia: botón de alarma	
Prioridad de negocio: alta	**Riesgo en desarrollo:** media
Iteración asignada: 1	
Responsable: Carlos Montesdeoca	
Descripción: el usuario del sistema puede acceder al aplicativo y usar un botón de pánico que active una alarma.	
Observación: La pulsación o activación del botón emitirá una alarma desde el dispositivo móvil.	

Product Backlog

El listado de todas las actividades clasificadas por su funcionalidad se presenta en esta sección donde su orden depende de la importancia del negocio o proyecto donde se aplique [17]. el listado realizado para este proyecto se puede encontrar en el manual técnico página (5).

Sprint Backlog

Otro listado de actividades, pero con una estimación de tiempo que va de la mano con las tareas definidas en el Product Backlog[17]. Una vez identificados y definidos los requerimientos que tendrá el sistema se estableció 5 Sprint contenidos en diferentes módulos de desarrollo, los cuales están redactados en el **Anexo** II en la sección **Sprint Backlog**.

Diseño de interfaces

Para un desarrollo web se unos diseños gráficos denominados (mockups). Su función principal es dar una idea tanto al equipo desarrollador como a los clientes una noción de lo que se está por realizar, con ayuda de la herramienta Figma se desarrolló los prototipos que ayudaran a tener una plantilla guía para la estructuración grafica del aplicativo.

En la *Fig 2.* se presenta un ejemplo de uno de los prototipos destinados al aplicativo. Para el resto de los prototipos se puede visualizar en el **Anexo II** en la sección de **Prototipos Gráficos.**

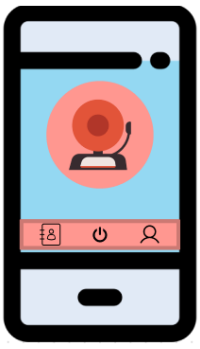


Fig 2. Interfaz gráfica pagina 2 botón de pánico

Arquitectura del sistema

Ya definidos los Sprints con los que se llevara a cabo el proyecto hasta su cierre, se creó el patrón arquitectónico para dar una vista global de cómo funciona el sistema.

Patrón arquitectónico

La modelo vista controlador (MVC) es bastante adecuado para este sistema, ya que el mismo distribuye las actividades del sistema en diferentes capas para crear una integridad y organización entre las mismas, facilitando su comprensión y mantenimiento [19].

La *Fig 3.* muestra el diseño que se realizó en esta etapa del proyecto y las herramientas que se usaron para el desarrollo del aplicativo móvil.

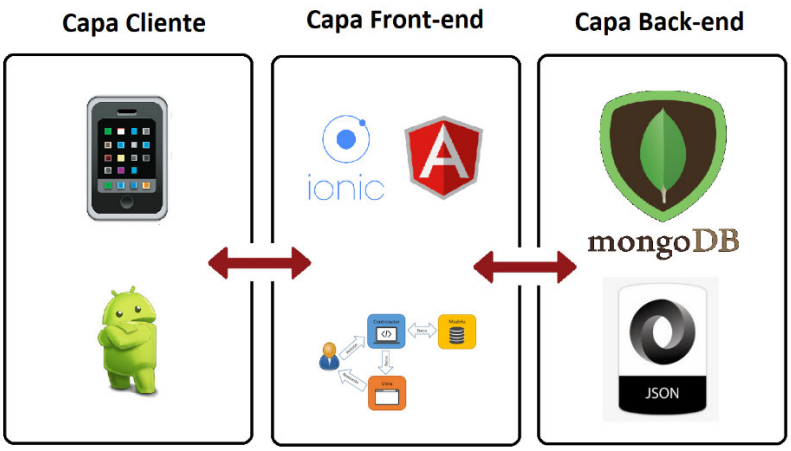


Fig 3. Patrón arquitectónico de aplicación móvil.

El desarrollo del sistema está basado en MEAN que por sus siglas se define como un servicio conformado por MongoDB, Express, Angular y Node. De esta forma se creó un desarrollo full-stack donde se cubren todos los servicios básicos de un aplicativo web, como los son front-end y back-end

MongoDB

Es un tipo de base de datos NoSQL bastante amigable con entornos móviles y web, la ventaja y gran diferencia de esta base de datos a comparación de otras del mismo tipo, es que esta tiene la facilidad de almacenar diferentes tipos de datos de forma mucho más organizada dentro de las colecciones que se crean dentro de la base de datos.

La razón para usar esta base de datos dentro del proyecto es que al usar su servicio como local la accesibilidad de datos y rapidez al momento de consultarlos es más efectiva.

Express

Express es la herramienta ideal al momento de crear servicios para un nuevo sistema, se lo conoce como un *web application framework*, ya que es un pequeño sistema que permite crear los servicios del lado del servidor junto con node de forma rápida y fácil.

Para este proyecto express es una excelente herramienta ya que al no ser un framework tan extenso y complicado facilita y agiliza la creación de los modelos y rutas que se usaran para la lógica del aplicativo.

Angular

Es un framework de desarrollo usado para la codificación del Front-end de las aplicaciones web o móviles.

El uso de este framework en este proyecto viene por la compatibilidad que tiene con Ionic ya que dentro de sus configuraciones iniciales permite crear un entorno con React, Angular o View, y en este caso se escogió Angular con una platilla de “Tabs” para agilizar el proceso de desarrollo y organización.

Node

Al mismo nivel del servidor junto con express se encuentra, Node ya que ayudara a gestionar todos los servicios que se creen y configuren a través de express para poder correrlo con JavaScript

Para una comprensión más fácil de la interacción de un MEAN Stack se presenta una ilustración con la participación de un cliente en la *Fig 4.*

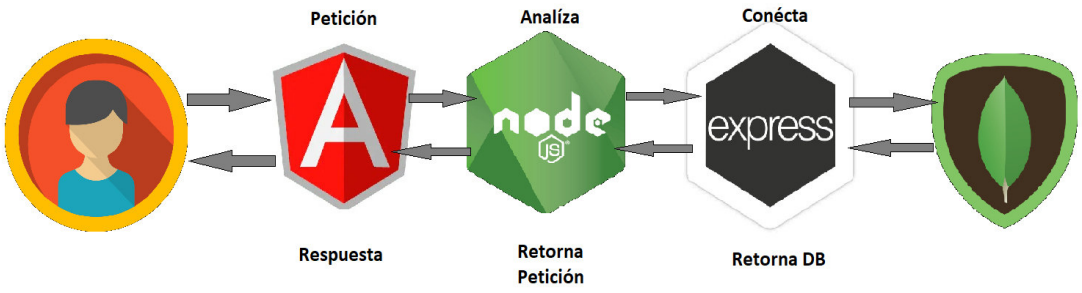


Fig 4. MEAN Stack

Herramientas del sistema

Las herramientas escogidas para el desarrollo de la aplicación están pensadas en facilitar y agilizar el proceso de desarrollo además de tener una vista futura de los posibles cambio o adaptaciones que puedan surgir una vez culminado este proyecto.

El listado detallado y justificado de las herramientas usadas se encuentra en la **TABLA II.**

TABLA II. Herramientas del sistema

Herramientas	Justificación
MongoDB	Este tipo de base de datos permite crear una estructura organizada sin la restricción de una estructura establecida como las bases de datos SQL, además de presentar un servicio en la nube que facilitara adaptar el proyecto en nuevas versiones
Angular	Es un framework desarrollado por Google que está destinado a crear aplicaciones web dentro de una sola página o web SPA (*Single Page Application).*
Ionic	Es un framework SDK de Front-end para el Desarrollo de aplicaciones hibridas basadas en tecnologías web con (HTML, CSS, JavaScript).
Bootstrap	Es un framework de interfaz de código abierto que permite agilizar y facilitar el proceso de creación de proyectos.
CSS	Es una configuración de estilos pata HTML que permite personalizar y agregar diferentes propiedades gráficas y dinámicas a las etiquetas.
Postman	Es una herramienta que se usa para hacer un testing de las API REST.

Para la creación y desarrollo del Back-end en el sistema MEAN Stack es necesario instalar algunas librerías que facilitaran el manejo y la interacción con el Front-end de la aplicación, a continuación, se presenta el listado detallado y justificado en la **TABLA III.**

TABLA III. Librerías para sistema MEAN Stack – Back-end

Librería	Descripción
Mongoose	Es una librería de node.js que está totalmente destinado al uso de una conexión con una base de datos MongoDB, esta permite también definir esquemas para estructurar y controlar el ingreso de información a la base de datos [13]. Aunque MongoDB no necesite un esquema para su almacenamiento es importante definirlo para la integridad y estandarización de la información que se usara.
Express	Es una librería de node que permite manejar peticiones, establecer ajustes de aplicación web como puertos de acceso y peticiones widdleware.
Parse-body	Es una librería de npm que permite visualizar y controlar la información enviada a través de un método POST o un PUT al sistema.
Cors	(*Cross Origin Resourse Sharing)* es el mecanismo con el que se envía un encabezado HTTP para dar permisos de acceso a un usuario a un dominio.
Nodemon	Es una librería destinada al desarrollo y testing de los servicios credos con node, la ventaja de su uso es poder crear un script donde se levantará el servidor y al realizar un cambio en el mismo, el servidor se apagará e iniciará automáticamente para corregir los cambios.

Twilio	Librería que permite crear notificaciones a través de envió de SMS.

3 RESULTADOS, CONCLUSIONES Y RECOMENDACIONES

3.1 Resultados

En la siguiente sección se puede evidenciar las tareas realizadas a lo largo del desarrollo del proyecto y los resultados de los Sprint definidos.

Sprint 0. Configuración del ambiente de desarrollo

Las tareas realizadas en el Sprint 0 son las siguientes:

- Diseño y creación de la base de datos
- Creación y configuración del proyecto con Ionic

Diseño y creación de la base de datos

El proyecto consume una base de datos tipo NoSQL con el servicio de MongoDB, en esta se almacenarán los datos como lista de contactos y las configuraciones que haga el usuario como el nombre con el que se enviara las notificaciones y el mensaje por defecto que puede ser actualizado, y para poder tener una visualización de los datos que se estén integrando o modificando se usara el servicio de MongoDB Compass. Para poder visualizar las colecciones y contenido de estas configuradas se puede chequear el **Anexo II** en la sección **Diseño de Base de Datos.**

Creación y configuración del proyecto en Ionic

Para la creación del proyecto se usó Node.js y Ionic Framework ya que los mismos permiten crear una plantilla inicial con las configuraciones básicas del framework, y para este proyecto fue necesario empezar con una plantilla de tablas, ya que esta creo las diferentes rutas y botones de navegación del sistema, así agilizando esta etapa de creación, en la *Fig 5.* se presenta la estructura inicial del proyecto una vez creado más una carpeta adicional donde se alojará el servidor del back-end.

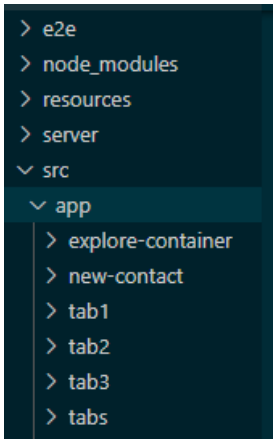


Fig 5. Estructura de directorios del aplicativo móvil

además de la creación y configuración automática que realiza automáticamente el generador de paletillas de Ionic se configuro la autoría del proyecto en el package.json, archivo que se encuentra en la raíz del proyecto.

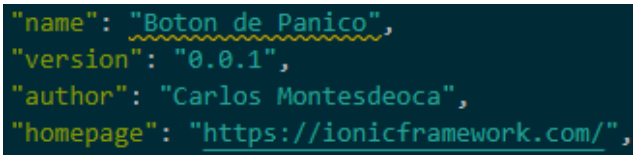


Fig 6. Configuración de autor del proyecto

Sprint 1. Módulo de usuario

Las tareas realizadas en el Sprint 1 son las siguientes:

- Creación de acceso rápido a la aplicación.
- Conexión con la base de datos

Creación de acceso rápido a la aplicación

Para la facilidad del acceso a la aplicación se optó por no usar una página de Log In como se está acostumbrado, ya que en este caso puede ser algo incensario tomando en cuenta que la base de datos no se va a manejar en la nube, de esta forma al abrir el aplicativo se dirigirá directamente a la página principal donde está la función principal del sistema, usando la plantilla ya generada por Ionic se usa la tab1 como directorio donde configurará dicha función.

Conexión con la base de datos

la conexión de la base datos está dada por un servidor creado con un modelo MEAN Stack alojado en el directorio servidor en la raíz del proyecto, al igual que con otros framework como Laravel que permiten crear plantillas para un Back-end, aquí se creó la plantilla desde cero, definiendo sus rutas, modelos y su conexión con MongoDB.

Al igual que el Front-end de la aplicación el Back-end que se creo tiene su propio archivo package.json, donde se agregaron las librerías necesarias para crear el sistema MEAN.

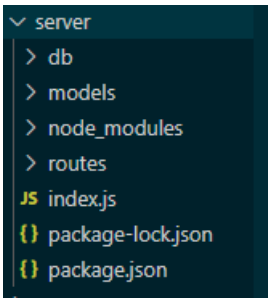


Fig 7. Estructura de back-end

```
const mongoose = require('mongoose');

const URI = 'mongodb://localhost/Apps';

mongoose.connect(URI)
  .then(db => console.log('Db is connected'))
  .catch(error => console.error(error));

module.exports = mongoose;
```

Fig 8. Conexión con MongoDB

También es importante definir el formato como se irán ingresando los datos a la base cuando el usuario interactúe con la aplicación, en la *Fig 9.* Se puede ver un ejemplo de un esquema de cómo se ingresarán los datos de los contactos.

```
const mongoose = require('mongoose');
const Schema = mongoose.Schema;

const ContactSchema = new Schema({
    name: { type: String },
    phone: { type: String }
})

module.export = mongoose.model('contactos', ContactSchema);
```

Fig 9. Esquema de contactos del usuario

Para probar la funcionalidad y verificar el acceso a la base de datos se hicieron algunas pruebas en Postman, enviado peticiones de tipo GET, POST, PUT y DELETE, y a su vez verificando que los cambios en la base de datos se vean reflejados.

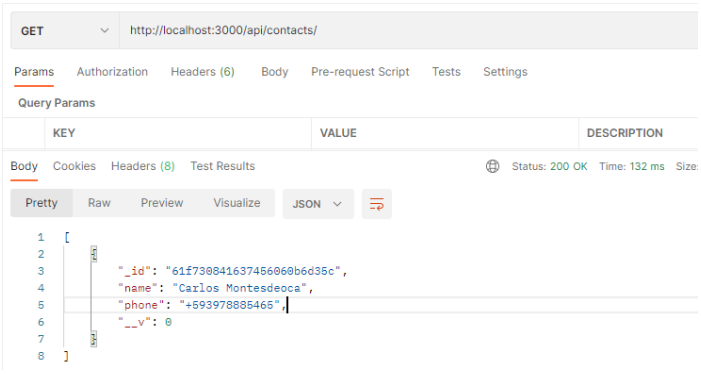


Fig 10. Pruebas de lectura (GET) en Postman

Una vez creado el back-end del sistema se importó las librerías necesarias para la lectura y escritura en la base de datos, también se crea un servicio donde se definen las rutas que se podrán usar para obtener los datos y presentarlos al usuario en forma de una lista tomando en cuenta los contactos como se presenta en la *Fig 10.*

Una vez ingresado a la sección de contactos el usuario solo tendrá la opción de eliminar o crear uno nuevo, y en la página de configuración se presenta un pequeño formulario para llenar con información personal y un mensaje personalizado.

Es importante recalcar que en esta parte el sistema detecta si es que el dato como nombre y mensaje no existen crea automáticamente información por defecto que debe modificarse, esto con el fin de evitar crear más registros de los necesarios.

Sprint 2. Módulo de alarma

Las tareas realizadas en el Sprint 2 son las siguientes:

- Creación de un botón de alarma de uso fácil e intuitivo.
- Configuración de una alarma sonora.
- Configuración y permisos para uso de librerías de GPS
- Vincular el botón de alarma con la función de GPS

- Configurar la extracción y lectura del GPS

Creación de un botón de alarma de uso fácil e intuitivo

El botón de pánico tiene que ser lo más sencillo y accesible posible ya que su uso es en emergencias por lo que la pagina donde está alojada la función simplemente carece de opciones de texto, configuraciones, anuncios u otra distracción. Por tal motivo la pantalla solo mostrará un botón grande que accione la función de ayuda, está pensado así ya que al ser un botón grande y centrado el usuario podrá accionarlo sin necesidad de visualizar la pantalla con se ve en la *Fig 11*.

Fig 11. Botón de pánico de la App

Configuración de una alarma sonora

El botón de pánico tiene la función de activar una alarma sonora igual que una alarma contra robos de una casa, obviamente con mucha menos potencia por la diferencia de capacidades de los dispositivos comparados, pero para el uso de esta en el sistema el archivo en formato MP3 está alojado dentro del proyecto.

Para la lectura del archivo .MP4 del sistema es necesario guardar el archivo que contiene la alarma de pánico, y a su vez definir una interfaz y modelo de datos con los que se

extraerá y leerá los datos de la alarma y posteriormente se exportará el archivo tab1.page.ts.

```
playAlarm (alarma: Alarma) {

  let audio = new Audio();

  audio.src = alarma.audio;
  alarma.reproduciendo = true;
  audio.load();
  audio.play();

  setTimeout(() => {
    alarma.reproduciendo = false;
  }, alarma.duracion * 100000);

  // () => this.geoInformation()
}
```

Fig 12. Función para reproducción de alarma

Configuración y permisos para uso de librerías de GPS

Para poder usar la geolocalización en la aplicación es necesario descargar la librería respectiva descrita en la **TABLA III** en la página (13). También se debe importar en la carpeta app.module.ts como se ve en la *Fig 13.* y configurarlo en el archivo tab1.page.ts donde está la función que se vinculara al botón.

```
import { HttpClientModule } from '@angular/common/http';
import { NativeGeocoder } from '@ionic-native/native-geocoder/ngx';
import { Geolocation } from '@ionic-native/geolocation/ngx';

@NgModule({
  declarations: [AppComponent],
  entryComponents: [],
  imports: [
    BrowserModule,
    IonicModule.forRoot(),
    AppRoutingModule,
    HttpClientModule // module
  ],
  providers: [
    NativeGeocoder,
    Geolocation,
    { provide: RouteReuseStrategy, useClass: IonicRouteStrategy }
  ],
  bootstrap: [AppComponent],
})
```

Fig 13. importación de las librerías de geolocalización en app.module.ts

Vincular el botón de alarma con la función de GPS

Una vez importada la librería que consulta la localización del usuario se crear la función que lea el archivo importado y lo reproduzca a través de la interfaz gráfica como se ve en la *Fig 14*.

```
geoInformation() {
  this.geolocation.getCurrentPosition().then((data) => {
    this.lat = data.coords.latitude;
    this.long = data.coords.longitude;
    this.accuracy = data.coords.accuracy;
    console.log(this.lat, this.long)
    console.log('https://www.google.com.ec/maps/@{{this.lat}},{{this.long}},17z?hl=es')
   }).catch((error) => {
    alert(error);
   });
}
```

Fig 14. función para geolocalización

Para llamar a la función de geoInformation se usa una nueva función llamada action, la cual llama a geoInformatio dentro de PlayAlarm, esto se hace de forma asíncrona para evitar una pérdida de tiempo esperando que la alama acabe de reproducirse para continuar con la obtención de la localización.

```
  async playAlarm (alarma: Alarma) {

    let audio = new Audio();

    audio.src = alarma.audio;
    alarma.reproduciendo = true;
    audio.load();
    audio.play();

    setTimeout(() => {
      alarma.reproduciendo = false;
    },
  alarma.duracion * 100000);
  const loc = await this.action();
}
```

Fig 15. Función múltiple del botón principal

Configurar la extracción y lectura del GPS

Adicionalmente para poder leer los datos de la ubicación se establece las variables donde se guardará la longitud y latitud del usuario, de esta forma esos datos se comparten con otros usuarios.

También se puede crear un enlace a través de Google mapa que direcciones a la ubicación usando las coordenadas obtenidas por el sistema, en la *Fig 16.* se muestra como la aplicación muestra los datos en consola.

```
▾ 0:
    cords: "https://www.google.com.ec/maps/@-0.28833536535053517,-78.49023375909123,17z?hl=es"
```

Fig 16. Enlace para localización de Google maps

Sprint 3. Módulo de notificación

Las tareas realizadas en el Sprint 3 son las siguientes:

- Crear una sección de contactos del usuario.
- Vincular los contactos para envió de SMS.
- Crear un mensaje personalizado para él envió de geolocalización.
- Vincular los datos de la base para enviar a los contactos del usuario.

Crear una sección de contactos del usuario

La sección de contactos del usuario esta manejada en la tab2. Creada por Ionic, en esta página el usuario puede ver un listado de todos sus contactos y a su vez crear uno nuevo, la información necesaria para crear el contacto es “el nombre del usuario y su número de teléfono.

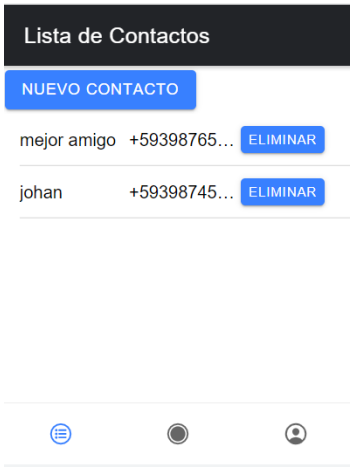


Fig 17. Página de lista de contactos

Vincular los contactos para envió de SMS

Para poder enviar los mensajes se debe leer toda la información de la configuración del usuario y de la lista de contactos del usuario, para lo que se crea una función llamada getMessage, la cual se encarga de crear la estructura que será enviada para los SMS como se ve en la *Fig 18*.

```
getMessages () {
  this.geoInformation();
  this.api.getConfig().subscribe(data => {
    this.user = data[0].name;
    this.text = data[0].message;
    this.from = data[0].phone;
  })

  this.api.getContacts().subscribe(data => {
    this.contacts = data;
  })

  for ( let i in this.contacts) {
    this.to = this.contacts[i].phone
    this.message = {
      body: `${this.text}. \n esta es mi ubicacion ${this.cord}`,
      from: this.from,
      to: this.to
    }
    this.api.newSMS(this.message).subscribe(response => {
      console.log(response);
    },
    error => {
      console.log(error);
    })
    this.listMessages.push(this.message)
```

Fig 18. Función getMessage

Para enviar los SMS a los contactos es importante definir tres datos de entrada, los cuales son solicitados por la librería de twilio que es gestionada en el back-end.

Para poder enviar un mensaje se usan los campos:

- to: que es el campo donde se guarda el contacto de destino.
- from: es el número del usuario.
- body: es el cuerpo del mensaje que será una cadena formada por los campos del mensaje personalizado y las coordenadas obtenidas por la función de geoInformation.

Como se ven en la *Fig 19*. newSMS es un servicio que gestiona la comunicación con la API a través de un método POST, pero es solo una simulación ya que en realidad lo envía la estructura del objeto a la base de datos, más bien se comunica con el servicio de Twilio para el uso de mensajería vía SMS.

```
router.post('/', async (req,res) => {

    const { from, to, body } = req.body;
    client.messages.create({
        body, from, to
    })

    .then(message => console.log(message.sid))
    res.json({status: 'mensaje guardado'});

});
```

Fig 19. Envió de SMS a través de Twilio

También hay que mencionar que la función getMessage se gestiona una vez acabe la obtención de las coordenadas por lo que se la incluye en la misma función action de la *Fig 16.* De allí venia la necesidad de crear una tercera función que gestione la localización y envió de SMS.

Crear un mensaje personalizado para él envió de geolocalización

El mensaje personalizado del usuario se encuentra en la tab3 del sistema, donde se encuentra toda la configuración necesaria para que el usuario pueda usar la aplicación, aunque esta configuración no es necesaria ya que por defecto ya tendrá un mensaje, la información que el usuario puede ingresar es el nombre de usuario y su mensaje si es que así lo desea, esto con el fin de facilitar identificar quien envía el mensaje.

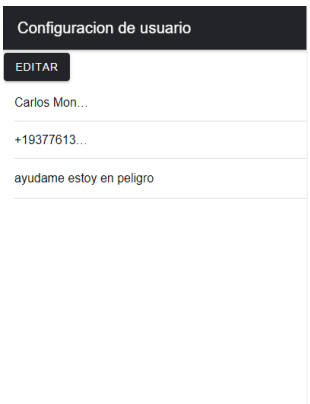


Fig 20. Página de información del usuario

Sprint 4. Módulo de pruebas

Las tareas realizadas en el Sprint 5 corresponden a todas las pruebas que el sistema debe pasar para su proceso de validación antes de usarlo en producción.

Pruebas Unitarias

Son los tipos de prueba que ayudan al equipo desarrollador a identificar los posibles problemas que puedan surgir probando bloques de código de forma individual, normalmente estos test están designados a probar una funcionalidad en concreto.

En base a las necesidades del aplicativo los test se realizaron en el servicio en controla todo lo relacionado al manejo de la base de datos y él envió de mensajes.

En la *fig 22*. Se puede ver un ejemplo del código para crear un test unitario con ayuda de Karma, que ya viene instalado y configurado en Ionic por defecto, y en la *Fig 23*. Se puede ver el resultado en consola una vez realizado el test.

```
it('send sms', (done: DoneFn) => {
  const sms = {
    "from": "+5987456321",
    "to": "+5987456321",
    "body": "message"
  }

  const resjson = {
    status:  'mensaje guardado'
  }
  httpClientSpy.post.and.returnValue(of(resjson));

  service.newSMS(sms)
  .subscribe( result => {
    expect(result).toEqual(resjson)
    done()
  })
})
```

Fig 21. Pruebas a sms

La *Fig 22.* Muestra cómo se simula él envió de datos necesario para que la librería de twilio pueda envíos los mensajes.

```
Chrome 97.0.4692.99 (Windows 10) ERROR
  Disconnected Client disconnected from CONNECTED state (transport close)
26 01 2022 18:20:30.567:INFO [Chrome 97.0.4692.99 (Windows 10)]: Connected on socket oFQiUoEAa4eONzpRAAA1 with id 66128043
Chrome 97.0.4692.99 (Windows 10): Executed 11 of 11 SUCCESS (0.238 secs / 0.206 secs)
TOTAL: 11 SUCCESS
```

fig 22. Resultados en consola

Pruebas de aceptación

La finalidad de estas pruebas es evaluar cómo se ha cumplido los requerimientos establecidos al inicio del sistema, para ello se usan las historias de usuario.

TABLA IV: Prueba de aceptación 2 - botón de alarma

HISTORIA DE USUARIO	
Identificador: PA-02	**Historia de usuario:** HU-02
Nombre de historia: botón de alarma	
Descripción: el usuario del sistema puede acceder al aplicativo y usar un botón de pánico que active una alarma.	
Condiciones de prueba: Acceso a internet	
Resultados: El usuario puede activar la alarma con solo presionar el botón central	

Pruebas de compatibilidad

En la **TABLA V.** se muestra un resultado del análisis en los tres entornos donde se probó el aplicativo móvil.

TABLA V. análisis pruebas de compatibilidad

Marca	Modelo	Evaluación
Xiaomi	Redmi 8	Aprobado
Huawei	P20 – Lite	Aprobado
Emulador	BlueStacks	Aprobado

Las evidencias en base a capturas de esta prueba están documentadas con graficas en los sistemas probados en el **Anexo II** en la sección de

Despliegue

Para el despliegue se escogió Heroku ya que da facilidad de crear un servidor en la nube de las mismas formas como se crea un repositorio de github, además se usó un cluster de mongoDB, ya que por defecto Heroku no admite conexiones locales y consultas.

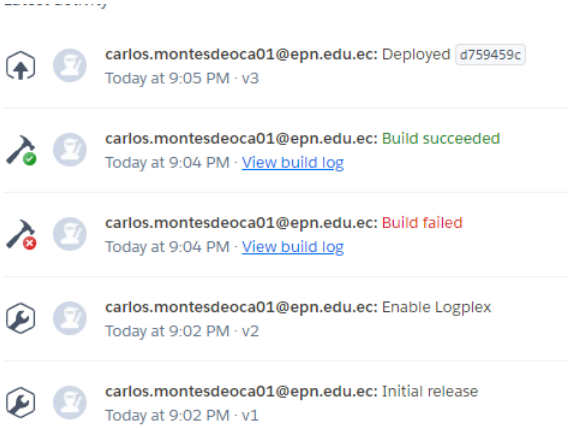


Fig 23. Estado del servidor back-end en Heroku

También es necesario realizar pruebas con las peticiones al servidor además de evidenciar en la terminal de Heroku el estado, por lo que se ha usado nuevamente el Postman para crear algunas peticiones como se ve en la Fig 24.

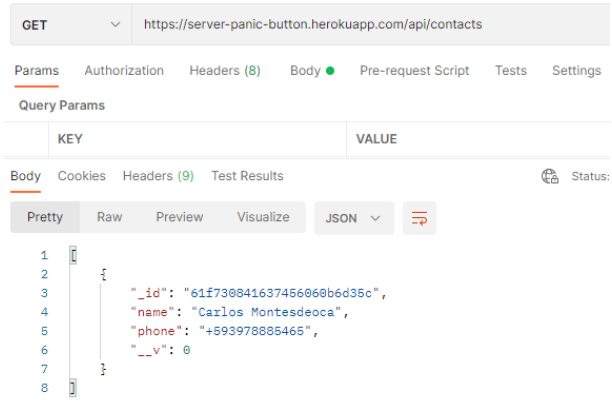


Fig 24. Pruebas Postman a servidor Heroku

Para el aplicativo móvil se usó la ayuda de las librerías incorporadas en Ionic como Capacitor/core que ayuda a generar el archivo build del aplicativo y posteriormente convertirlo a un proyecto java de Android.

El apk está disponible en el link de mega:

https://mega.nz/folder/U98VTQoL#v01BQk4MOxHSpQ3y6dePng

3.2 Conclusiones

- La aplicación permite gestionar los contactos y él envió de geolocalización exitosamente.
- Gracias a los servicios externos ofrecidos por empresas que también se dedican al desarrollo como el ejemplo más claro de este proyecto (twilio), se agilizo y mejoro el proceso de creación de diferentes aplicaciones o servicios web.
- La integración de varias tecnologías para el desarrollo de software es de gran importancia ya que permite evaluar este en diferentes entornos, y en ocasiones transformar el lenguaje inicial del programa en otro más amigable para otros entornos, como es el ejemplo del AndroidStudio que permitió crear un APK transformando el código a java.
- Node js es una parte muy importante que se debe enfatizar más en el aprendizaje ya que permite crear micro servicios y servidores desde lo más pequeño, lo cual facilita menormente el entendimiento a futuro cuando se crea un Back-end usando un framework más avanzado.

3.3 Recomendaciones

- Crear un apartado web que gestione las peticiones de alerta que arroje el aplicativo móvil con el fin de que esas notificaciones sean atendidas a la vez por el cuerpo autorizado de emergencia en este caso ECU 911.
- Configurar la librería de twilio para que los mensajes sean enviaos en la plataforma de WhatsApp, ya que tienen más uso y acogida por los usuarios.
- Configurar la función principal del sistema para que pueda ser activada sin necesidad de acceder a la aplicación.
- Crear una función que permita crear diferentes instancias en el clúster de mongoDB con la dirección única del dispositivo móvil para evitar registros o ingresos con autenticación.
- Crear un sistema que permita seguí en tiempo real el recorrido del dispositivo móvil una vez activada la alarma.

4 REFERENCIAS BIBLIOGRÁFICAS

[1]. f. alcívar mendoza, "la inseguridad ciudadana como factor de riesgo de bien público en la facultad de ingeniería industrial en la jornada nocturna de la universidad de guayaquil en el 2018", *repositorio.ug.edu.ec*, 2020. [online]. available: http://repositorio.ug.edu.ec/handle/redug/27512. [accessed: 08- dec- 2020].

[2]. k. fernández oñate, "video animado para prevenir el acoso callejero que sufren las mujeres en la ciudad de quito", *dspace.udla.edu.ec*, 2020. [online]. available: http://dspace.udla.edu.ec/handle/33000/11835. [accessed: 08- dec- 2020].

[3]. d. barreto, "respuesta de política pública frente al consumo", *200.41.82.22*, 2020. [online]. available: http://200.41.82.22/handle/10469/8360. [accessed: 08- dec- 2020].

[4]. j. pita cáceres, "inmigración en ecuador: incidencia en el mercado laboral en la ciudad de guayaquil.", *192.188.52.94*, 2020. [online]. available: http://192.188.52.94/handle/3317/11668. [accessed: 08- dec- 2020].

[5]. aguilera albán, "distribución espacial de delitos contra la propiedad y percepción de espacios urbanos inseguros en el barrio la floresta", *repositorio.puce.edu.ec*, 2020. [online]. available: http://repositorio.puce.edu.ec/handle/22000/16180. [accessed: 08- dec- 2020].

[6]. clay alvino, “estadísticas de la situación digital en ecuador 2021-2021, mayo 2021. [online]. available: https://branch.com.co/marketing-digital/estadisticas-de-la-situacion-digital-de-ecuador-en-el-2020-2021/#:~:text=el%20n%c3%bamero%20de%20dispositivos%20m%c3%b3viles,%2c8%25%20de%20la%20poblaci%c3%b3n.accessed: 08- dec- 2020].

[7]. fiscacalia general del estado, estadísticas fge, robos 2021. aviable: https://www.fiscalia.gob.ec/estadisticas-de-robos/. [accessed :28-nov-2021].

[8]. "la delincuencia tiene su 'nueva normalidad' y cuatro expertos la explican", *primicias*, 2020. [online]. available: https://www.primicias.ec/noticias/sociedad/delincuencia-nueva-normalidad-expertos-explican/. [accessed: 08- dec- 2020].

[9]. "twitter contra el delito en ecuador - bbc news mundo", *bbc news mundo*, 2020. [online]. available:

https://www.bbc.com/mundo/america_latina/2009/12/091229_0448-ecuador_twitter_gz. [accessed: 08- dec- 2020].

[10]. j. pontón cevallos and a. santillán, *nuevas problemáticas en seguridad ciudadana*. quito: flacso, 2010.

[11]. lima león and c. valdez bermejo, "diseño de un botón de pánico móvil para adultos mayores para el monitoreo a través del sistema ecu 911", *dspace.ups.edu.ec*, 2020. [online]. available: https://dspace.ups.edu.ec/handle/123456789/17180. [accessed: 07- dec- 2020].

[12]. j. andrade quizhpe, "análisis espacial de la distribución del delito de robo en el distrito metropolitano de quito-ecuador en el año 2017", *dspace.uce.edu.ec*, 2020. [online]. available: http://www.dspace.uce.edu.ec/handle/25000/18410. [accessed: 07- dec- 2020].

[13]. Jose Luis Coalla Cencerrado, universidad politécnica de Madrid, como crear un sistema con MEAN Stack, 2018.

[14]. e. burnette, *android*. madrid: anaya multimedia, 2011.

[15]. l. pineda soto, j. zucchet amaya and n. lópez giraldo, *metodologías ágiles de desarrollo*. bogotá: uniandes, 2007.

[16]. martel, gestión práctica de proyectos con scrum.

[17]. MANUEL TRIAS GALLEGO Y ANA CRISTINA GALLEGO T,METODOLIGA SCRUM.

[18]. h. ñaupas paitán, e. mejía mejía and e. novoa ramírez, metodología de la investigación cuantitativa-cualitativa y redacción de la tesis (4a. ed.). bogotá: ediciones de la u, 2014.

[19]. desarrolloweb.com, «desarrolloweb.com,» desarrolloweb.com, 28 julio 2020. [en línea]. available: https://desarrolloweb.com/articulos/que-es-mvc.html. [último acceso: 16 septiembre 2021].

[20]. la serana."estudio correlacional de factores como desempleo e índices de delincuencia en ecuador"2019. [online]available: https://scielo.conicyt.cl/scielo.php?pid=s0718-07642019000300287&script=sci_arttext&tlng=n. [accessed: 08- ene- 2021].

[21]. arcotel."internet movil".2020.[online]available: http://www.arcotel.gob.ec/464-de-usuarios-del-servicio-movil-avanzado-poseen-un-smartphone/ [accessed: 12- ene- 2021].

[22]. diario el expreso. (2014, febrero 20). la aplicación móvil para el ecu 911, funciona en quito. [en línea]. disponible en: http://www.expreso.ec/historico/laaplicacion-movil-para-el-ecu-911-funciona-kwgr_5750797

[23]. diario el expreso. (2019, julio 17). la metodología kanban para el desarrollo de procesos ágiles [en línea]. disponible en: https://hdl.handle.net/123456789/14948

[24]. mongoose docs, https://mongoosejs.com/docs/api.html

[25]. ian sommrville, *ingenieria de software* .pearson addison wesley, 2006.

[26]. java-script: the definitive guide, second edition david flanagan, and gregor m. novak

[27]. utilidad y funcionamiento de las bases de datos nosql databases nosql's utility and functioning, alexander castro romero, 2012

[28]. https://codigofacilito.com/articulos/que-es-mongoose

[29]. Codigo Front-end → https://github.com/CarlosMontesdeoca/boton-de-panico.git

[30]. Codigo Back-end → https://github.com/CarlosMontesdeoca/server-app.git

5 ANEXOS

- Anexo I.
- Anexo II.
- Anexo III.
- Anexo IV.

ANEXO II

MANUAL TECNICO DE LA APLICACIÓN MOVIL

1. Recopilación de requerimientos

En la **TABLA I.** se presenta el listado de requerimientos que el cliente solicita en el sistema.

TABLA I: Recopilación de requerimientos.

Recopilación de requerimientos		
Tipo de sistema	**ID - RR**	**Enunciado del ítem**
Aplicación Móvil	RR-01	Como usuario del sistema me gustaría tener un ingreso rápido a la app sin inicio de sesión.
	RR-02	Como usuario del sistema me gustaría una alarma al momento de activar la app.
	RR-03	Como usuario del sistema me gustaría enviar mi ubicación al momento de activar la app.
	RR-04	como usuario del sistema me gustaría poder guardar algunos números de mis contactos.
	RR-05	Como usuario del sistema me gustaría poder crear y editar un mensaje personalizado.
	RR-06	Como usuario del sistema me gustaría compartir mi ubicación con al momento de activar la app.

2. Historias de usuario

Como complemento a la recopilación de requerimientos se definen las historias de usuario que van desde la **TABLA II** hasta la **TABLA VIII,** en las siete historias de usuario se presenta una guía para el desarrollo de las diferentes funcionalidades del aplicativo móvil.

TABLA II: historia de usuario 1 - ingreso de aplicación

HISTORIA DE USUARIO	
Identificador: HU-01	**Usuario:** usuario
Nombre de historia: ingreso de aplicación	
Prioridad de negocio: media	**Riesgo en desarrollo:** media
Iteración asignada: 1	
Responsable: Carlos Montesdeoca	
Descripción: El usuario del sistema necesita un ingreso rápido al sistema	
Observación: El usuario necesita un acceso rápido a la aplicación sin necesidad de inicios de sesión continuas cada vez que se quiera usar la aplicación móvil.	

TABLA III: historia de usuario 2 - botón de alarma

HISTORIA DE USUARIO	
Identificador: HU-02	**Usuario:** usuario
Nombre de historia: botón de alarma	
Prioridad de negocio: alta	**Riesgo en desarrollo:** media
Iteración asignada: 1	
Responsable: Carlos Montesdeoca	
Descripción: el usuario del sistema puede acceder al aplicativo y usar un botón de pánico que active una alarma.	
Observación: La pulsación o activación del botón emitirá una alarma desde el dispositivo móvil.	

TABLA IV: historia de usuario 3 - activar geolocalización

HISTORIA DE USUARIO	
Identificador: HU-03	**Usuario:** usuario
Nombre de historia: activar geolocalización	
Prioridad de negocio: alta	**Riesgo en desarrollo:** alta
Iteración asignada: 2	
Responsable: Carlos Montesdeoca	
Descripción: El usuario puede extraer la geolocalización en tiempo real cuando se active el botón de pánico.	
Observación: La geolocalización se obtendrá cuando el usuario active el botón principal de la aplicación móvil.	

TABLA V: historia de usuario 4 – contactos preferidos del usuario

HISTORIA DE USUARIO	
Identificador: HU-04	**Usuario:** usuario
Nombre de historia: contactos preferidos del usuario	
Prioridad de negocio: media	**Riesgo en desarrollo:** media
Iteración asignada: 2	
Responsable: Carlos Montesdeoca	
Descripción: El usuario puede crear una agenda de contactos preferidos o de confianza.	
Observación: Debe existir un apartado donde el usuario pueda crear una pequeña agenda con usuarios de confianza a quienes notificar una situación de peligro.	

TABLA VI: historia de usuario 5 – mensaje de ayuda personalizado

HISTORIA DE USUARIO	
Identificador: HU-05	**Usuario:** usuario
Nombre de historia: mensaje de ayuda personalizado	
Prioridad de negocio: baja	**Riesgo en desarrollo:** baja
Iteración asignada: 1	
Responsable: Carlos Montesdeoca	
Descripción: El usuario puede crear un mensaje personalizado. El usuario puede editar el mensaje creado.	
Observación: La aplicación puede enviar un mensaje por defecto con la geolocalización o un mensaje personalizado con la geolocalización.	

TABLA VII: historia de usuario 7 – envió de geolocalización

HISTORIA DE USUARIO	
Identificador: HU-06	**Usuario:** usuario
Nombre de historia: Envío de geolocalización	
Prioridad de negocio: alta	**Riesgo en desarrollo:** alta
Iteración asignada: 2	
Responsable: Carlos Montesdeoca	
Descripción: el usuario puede compartir la ubicación junto con el mensaje personalizado si es que se creó a los contactos preferidos definidos anteriormente.	
Observación: La geolocalización debe ser enviada a los contactos de confianza configurados por el usuario.	

3. Product Backlog

La priorización de cada una de las funcionalidades que se implementan en el aplicativo móvil es presentada en orden en la **TABLA IX.**

TABLA VIII: Product backlog.

ELABORACION DE PRODUCT BACKLOG				
ID-HU	**Historia de usuario**	**Iteraciones**	**Estado**	**Prioridad**
HU-01	Ingreso de aplicación	1	Pendiente	Media
HU-02	Botón de alarma	1	Pendiente	Alta
HU-03	Activar geolocalización	2	Pendiente	Alta
HU-04	Contactos preferidos del usuario	2	Pendiente	Media
HU-05	Mensaje de ayuda personalizado	1	Pendiente	Baja
HU-06	Envío de geolocalización	2	Pendiente	Alta

4. Sprint Backlog

En la **TABLA X.** se muestra la creación de los diferentes Sprints que se llevaran a cabo para la creación del proyecto, tomando en cuenta la división de módulos según la funcionalidad asignada al sistema y una estimación de tiempos para cada actividad.

TABLA IX: Sprint backlog.

ELABORACION DE SPRINT BACKLOG					
ID-SB	**NOMBRE**	**ID-HU**	**HISTORIA DE USUARIO**	**TAREAS**	**ESTIMACION**
SB-01	configuración de entorno de desarrollo	N/A	N/A	Creación de base de datos NoSQL. Creación de proyecto con framework en Ionic.	10H
SB-02	Módulo de Usuario	HU-01	Ingreso de aplicación	Crear un acceso a la aplicación rápida. Conexión con la base de datos.	10H
SB-03	Módulo de alarma	HU-02	Botón de alarma	Crear un botón de alarma con un uso fácil e intuitivo. Configurar la alarma sonora en la activación de la función principal.	30H
		HU-03	Activar geolocalización	Configurar los permisos y librerías respectivas para usar GPS del teléfono. Vincular la función de extracción de localización con el botón de alarma. Configurar formatos de extracción y lectura de localización.	40H

SB-04	Módulo de notificación	HU-04	Contactos preferidos del usuario	Crear una sección donde el usuario pueda agregar contactos del teléfono. Vincular los contactos con una función de envió de SMS. Configurar el botón de alarma con los contactos del usuario.	25H
		HU-05	Mensaje de ayuda personalizado	Crear un mensaje personalizado para el envío de geolocalización. Configurar el formato de lectura y escritura para usar en la base de datos.	15H
		HU-07	Envío de geolocalización	Configuración de estructura de lectura de base de datos. Configuración de estructura de extracción de datos: - mensaje personalizado. - locación por GPS. Vincular envío de datos a los contactos del usuario.	40H
SB-05	Módulo de pruebas	N/A	N/A	Pruebas de funcionalidad al sistema. • Pruebas unitarias • Pruebas de rendimiento • Pruebas de aceptación	30H
Documentación					40H
Total de horas					240H

5. Prototipos Graficos

A continuación, se puede ver los prototipos desde la *fig1.* hasta la *fig3.* con los que sirvió como referencia para la creación del presente proyecto de titulación.

fig1. Interfaz gráfica página 1 "botón de pánico"

En *fig1.* Se muestra la página inicial del aplicativo el cual tiene únicamente el botón principal que activara la función de pánico, donde se incluirán las funciones de locación y notificación.

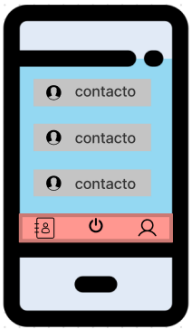


fig2. Interfaz gráfica página 2 "contactos"

En la *fig2.* Se muestra la segunda página del aplicativo, navegable desde la barra de tags o navegador, donde el usuario deberá configurar los contactos a notificar.

fig3. Interfaz gráfica página 3 “mensaje personalizado”

En la *fig3.* Se muestra la tercera página del aplicativo, navegable desde la barra de tags o navegador, donde el usuario tendrá la opción de crear un mensaje personalizado o dejar un mensaje por defecto de la aplicación.

6. Diseño de Base de Datos

En la *fig4*. Se ilustra la tabla con el modelo de base de datos para los contactos del usuario, mientras que la *fig5***.** Se muestran los datos que se usaran para configurar el perfil del usuario.

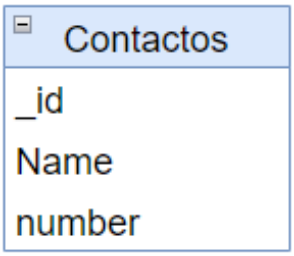


fig4. Colección para contactos del usuario.

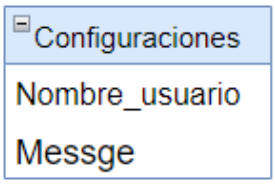


fig5. Colección para configuraciones del usuario.

7. Pruebas.

7.1 Pruebas unitarias

Para poder verificar las acciones que se ejecutan en el proyecto se ve la necesidad de realizar pruebas unitarias ya que así también se pueden encontrar errores o comportamientos inesperados en las funcionalidades por individual.

Las pruebas se han realizado analizando el servicio que controla todas las peticiones a la base de datos, ya que ahí se maneja toda la información importante y también se maneja las notificaciones vía SMS.

Como se evidencia en las fig6 a la fig8. Se crean diferentes tareas que pruebas cada método (post, put, delete) en la base de datos analizándola por cada colección, y agregando el método HttpClientTestinModule. Dicha importación ayudara a simular la conexión de una base de datos sin relacionarlo por cada componente.

```
it('create contact' , (done: DoneFn) => {
  const contact = {
    "name": "Carlos Montesdeoca",
    "phone": "+59978885465"
  }

  const resjson = {
    "status": "contacto guardado"
  }

  httpClientSpy.post.and.returnValue(of(resjson))

  service.newContact(contact)
    .subscribe( result => {
      expect(result).toEqual(resjson)
      done()
    })
  }
)

it('delete contact', (done: DoneFn) =>{
  const contact = "61f1b42e1434df4614b85920"

  const resjson = {
    "status": "contacto eliminado"
  }

  httpClientSpy.delete.and.returnValue(of(resjson))

  service.deleteContact(contact)
    .subscribe( result => {
      expect(result).toEqual(resjson)
      done()
    })
})
```

fig6. Prueba unitaria para contactos

En la fig6. Se muestran las pruebas realizadas en el servicio que controlo9 los contactos, se simula la forma en como deben ser enviados los datos del contacto y como responde el Api creado con node.

```
it('create config', (done: DoneFn) => {
  const config = {
    "name": "useer",
    "phone": "+59398754654",
    "message": "ayuda"
  }

  const resjson = {
    status: 'configuracion guardada'
  }

  httpClientSpy.post.and.returnValue(of(resjson));

  service.newConfig(config)
    .subscribe( result => {
      expect(result).toEqual(resjson)
      done()
    })
  }
)

it('edit config' , (done: DoneFn) => {
  const config = {
    "name": "useer",
    "phone": "+59398754654",
    "message": "ayuda"
  }

  const resjson = {
    status: 'configuracion actualizada'
  }
  httpClientSpy.put.and.returnValue(of(resjson));

  service.editConfig(config)
    .subscribe( result => {
      expect(result).toEqual(resjson)
      done()
    })
```

fig7. Prueba unitaria para configuracion

En la fig7. Se muestran las pruebas realizadas en el servicio que controlo9 los contactos, se simula la forma en como deben ser enviados los datos del contacto y como responde el Api creado con node.

Aun que intuitivamente el usuario no crea la configuracion inical del proyecto, este se ejecuta de forma automatica si es que el aplicativo detecta que no existe una en la base de datos, por tal motivo es que se ha incluido dentro de las pruebas.

```
it('send sms', (done: DoneFn) => {
  const sms = {
    "from": "+5987456321",
    "to": "+5987456321",
    "body": "message"
  }

  const resjson = {
    status:  'mensaje guardado'
  }
  httpClientSpy.post.and.returnValue(of(resjson));

  service.newSMS(sms)
  .subscribe( result => {
    expect(result).toEqual(resjson)
    done()
  })
})
```

fig8. Prueba unitaria para evio de sms

En la fig8. Se muestra la prueba realizada para la funcion de envio de mensajes, la que esta definida en la api como un metodo post, aun que enrealidad no guarda nada en la base dedatos, se lo usa de esta forma ya que necesita recibir los

parametros mostrados, pero enviadolos a un servicio externo que gestiona el envio de SMS.

Una vez realizadas las pruebas en cada componente se puede evidenciar en la pantalla de Karma que la apliacaion logro pasar, aun que se muestran mas componentes en la fig9. Estos son solo la prueba de carga de los diferentes compoentes.

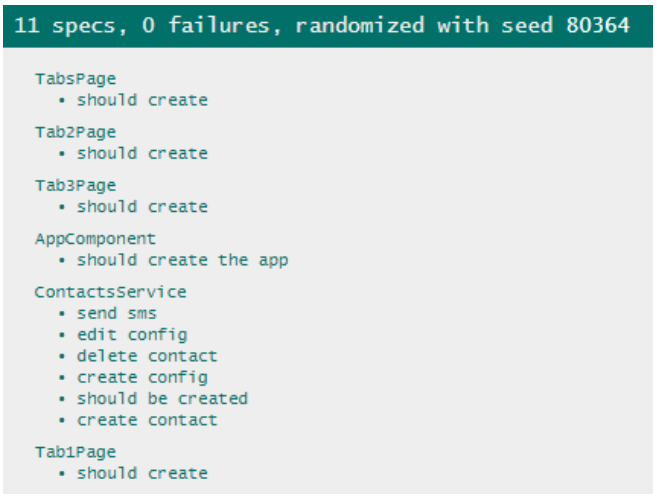


fig9. Resultados de Karma.

7.2 Pruebas de aceptación

En base a los requerimientos recopilados se procede a evaluar cada uno de los ítems descritos en las **TABLA XI** a la **TABLAXVI**

TABLA XI: Prueba de aceptación para historia de usuario HU-01

PRUEBA DE ACEPTACION	
Identificador: PA-01	**Historia de usuario:** HU-01
Nombre de historia: ingreso de aplicación	
Descripción: El usuario del sistema necesita un ingreso rápido al sistema	
Condiciones de prueba: Ninguna	
Pasos de ejecución: - Ingreso a la aplicación	
Resultados: La aplicación no demora en cargar los componentes e ingresa sin ningún registro o credenciales.	

TABLA XII: Prueba de aceptación para historia de usuario HU-02

HISTORIA DE USUARIO	
Identificador: PA-02	**Historia de usuario:** HU-02
Nombre de historia: botón de alarma	
Descripción: el usuario del sistema puede acceder al aplicativo y usar un botón de pánico que active una alarma.	
Condiciones de prueba: Acceso a internet	
Resultados: El usuario puede activar la alarma con solo presionar el botón central	

TABLA XIII: Prueba de aceptación para historia de usuario HU-03

HISTORIA DE USUARIO	
Identificador: PA-03	**Historia de usuario:** HU-03
Nombre de historia: activar geolocalización	
Descripción: El usuario puede extraer la geolocalización en tiempo real cuando se active el botón de pánico.	
Condiciones de Prueba: Ninguna	
Resultados: El aplicativo pide al usuario dar permisos de acceso al GPS y extrae las coordenadas para poder compartirlas con sus contentos	
Observaciones: El aplicativo de vez en cuando no extrae bien las coordenadas por lo que debe pulsarse dos veces el botón.	

TABLA XIV: Prueba de aceptación para historia de usuario HU-04

HISTORIA DE USUARIO	
Identificador: PA-04	**Historia de usuario:** HU-04
Nombre de historia: contactos preferidos del usuario	
Descripción: El usuario puede crear una agenda de contactos preferidos o de confianza.	
Condiciones de Prueba: Ninguna.	
Resultados: El usuario puede ingresar los contactos de su preferencia. Nota: el contacto se guarda en formato → +593987654321	

TABLA XV: Prueba de aceptación para historia de usuario HU-05

HISTORIA DE USUARIO	
Identificador: PA-05	**Historia de usuario:** HU-05
Nombre de historia: mensaje de ayuda personalizado	
Descripción: El usuario puede crear un mensaje personalizado. El usuario puede editar el mensaje creado.	
Condiciones de Prueba: Ninguna.	
Resultados: El usuario puede crear y editar su información en el apartado de configuración, para lo que es necesario que el usuario ingrese su número telefónico mensaje y su nombre Nota: el número no es un campo obligatorio sin embargo si no se ingresa la aplicación no tiene una funcionalidad.	

TABLA XVI: Prueba de aceptación para historia de usuario HU-06

HISTORIA DE USUARIO	
Identificador: HU-06	**Historia de usuario:** HU-06
Nombre de historia: Envío de geolocalización	
Descripción: el usuario puede compartir la ubicación junto con el mensaje personalizado si es que se creó a los contactos preferidos definidos anteriormente.	
Condiciones de Prueba: Acceso a internet.	
Resultados: La aplicación ingresa a su memoria las coordenadas y posteriormente procede a crear un listado de mensajes tomando los campos: From → que es el número del contacto, To → que es el número del usuario, Body → es el cuerpo del mensaje de texto que será enviado.	

7.3 Pruebas de compatibilidad

El aplicativo se lo probo en tres dispositivos físicos para probar su comportamiento y aspecto y en un emulador como se muestra en las fig10 a la fig12.

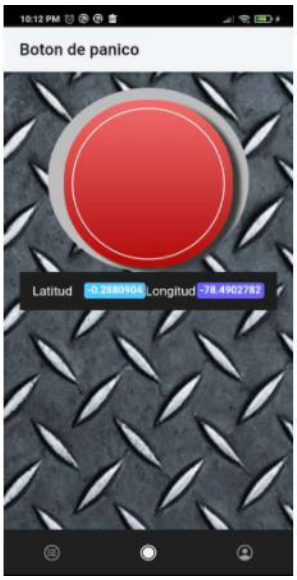


fig10. Prueba de compatibilidad xiaomi redmi 8.

fig11. Prueba de compatibilidad Huawei P20 lite.

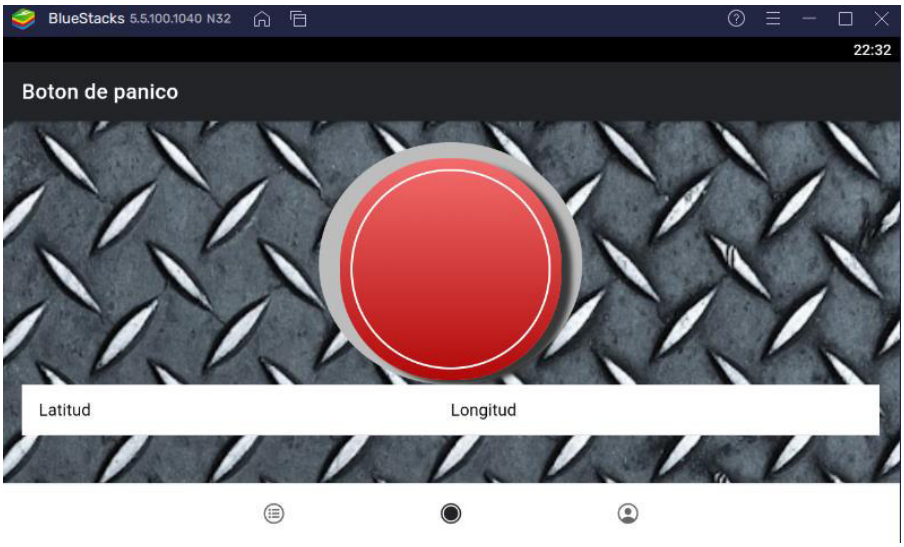


fig12. Prueba de compatibilidad emulador blueStacks.

ANEXO III

MANUAL DE USUARIO

COMO USAR BOTON DE PANICO

El aplicativo consta de 3 interfaces en donde se manejan todas las funcionalidades requeridas, lo primero que se verá es la pantalla de inicio con la función principal.

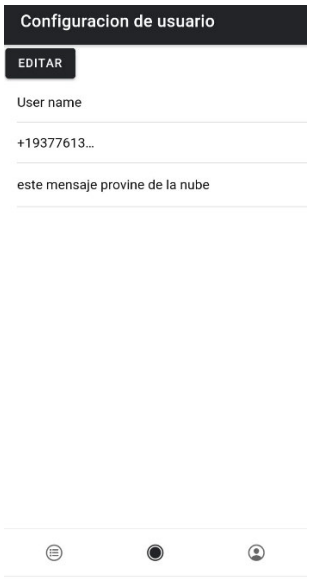
Los tags del 1 al 3 permiten navegar por la aplicación:

- 1: direcciona a el listado de contactos.
- 2: direcciona a la página principal del botón
- 3: direcciona a las configuraciones del usuario.

Los siguientes pasos permiten configurar la app para su uso.

Configuración del usuario

Al iniciar el aplicativo se lanza una función que evalúa si existen datos de configuración, en caso contrario aparecerá un numero por defecto, un nombre de usuario genérico y un mensaje por defecto.



Al entrar a la opción de editar información se despliega un pequeño formulario que pide el nombre del usuario, un mensaje personalizado y un número de teléfono que por defecto solo se puede con el código postal de Ecuador (+593).

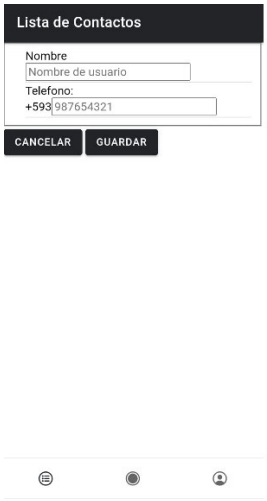
Se debe configurar con el numero perteneciente al usuario, ya que ese número enviara los SMS.

Lista de Contactos.

El listado de usuario aparecerá en el tag 2 y permite ingresar únicamente el nombre del contacto y su número en formato internacional.

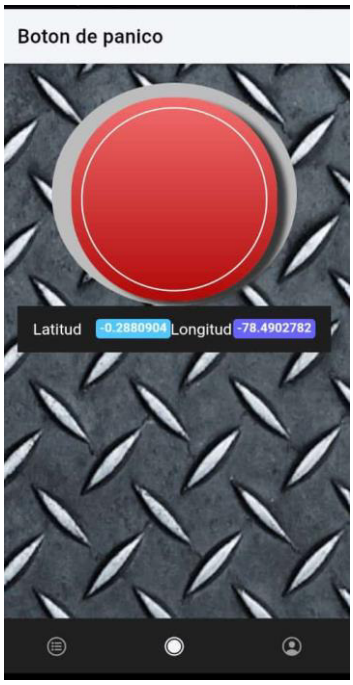
No se debe ingresar en el campo teléfono un numero empezando con el 0 por que el formulario no lo reconocerá, se debe ingresar como ejemplo (987654321).

El sistema internamente enviara el numero en formato internacional con el código de Ecuador.



Cuando se configure la información del usuario, y al menos un contacto la función del botón de pánico sacara el 100% de su provecho, al activar el botón internamente mapeara cada contacto, extraerá las coordenadas y enviara un mensaje a los contactos uno a uno.

Se puede evidenciar que el GPS funciona ya que pedirá los permisos respectivos y en la pantalla aparecerán los valores.



Nota Importante: Botón de pánico no es una salvación y emite una alarma que dependiendo de cada teléfono suena más o menos, si es que la situación lo amerita subir el volumen al máximo, caso contrario no arriesgar su integridad física.

ANEXO IV

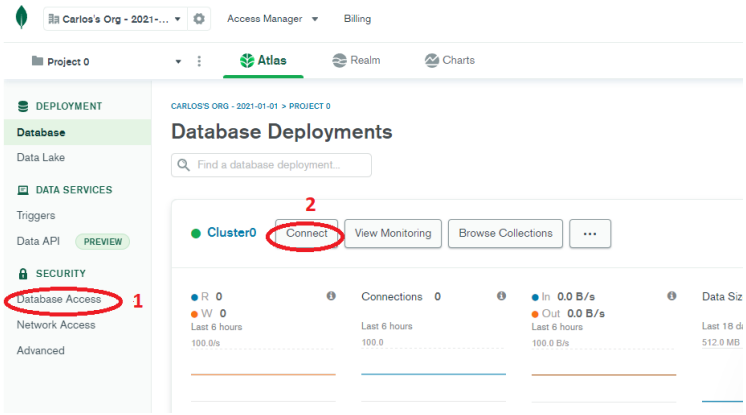
MANUAL DE INSATALACIÓN

Despliegue back-end

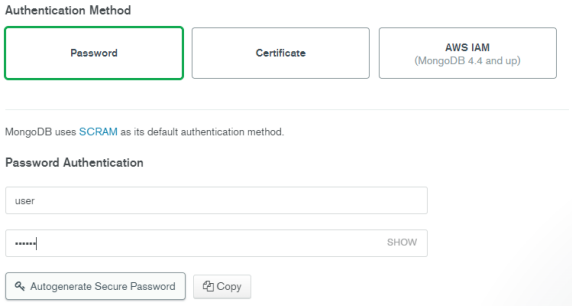
Para poder tener el aplicativo primero es necesario tener una cuenta en mongoDB Atlas para tener un cluster donde se administren los datos ingresados.

Para crear el cluster primero se dirige a la página de mongoDB Atlas, registrarse y crear una nueva base de datos.

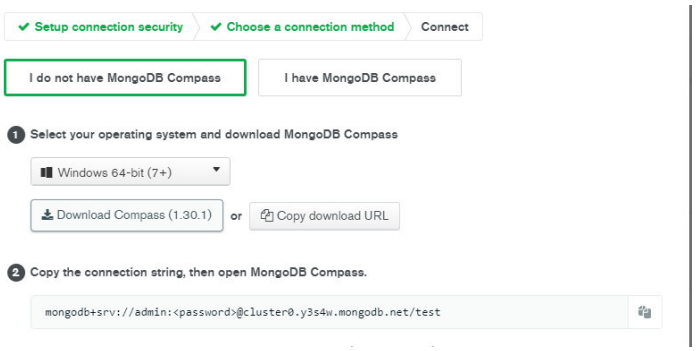
Ahora se debe crear un usuario y contraseña en la interfaz de mongoDB Atlas, ahora con esas credenciales de acceso, es necesario cambiarlos en el código del back-end.



En el punto uno se debe crear un usuario y contraseña para el mismo, el cual va a tener permisos de lectura y escritura.



Ahora para poder definir una ruta de conexión al clúster de mongodb se debe dirigir al punto dos, ahí saldrá la cadena de texto que se debe usar para poder conectarse a la base de datos.



Hay que tener en cuenta que el campo <password> se debe remplazar por la contraseña que se ingresó al crear el usuario.

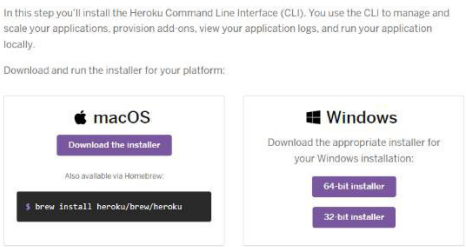
Esta cadena de texto se debe ingresar en el proyecto back-end de botón de pánico dentro del archivo database.js

```
src > JS database.js > URI
1   const mongoose = require('mongoose');
2
3   const URI = 'mongodb+srv://admin:<password>@cluster0.y3s4w.mongodb.net/test';
4
5   mongoose.connect(URI)
6     .then(db => console.log('Db is connected'))
7     .catch(error => console.error(error));
8
9   module.exports = mongoose;
```

Heroku

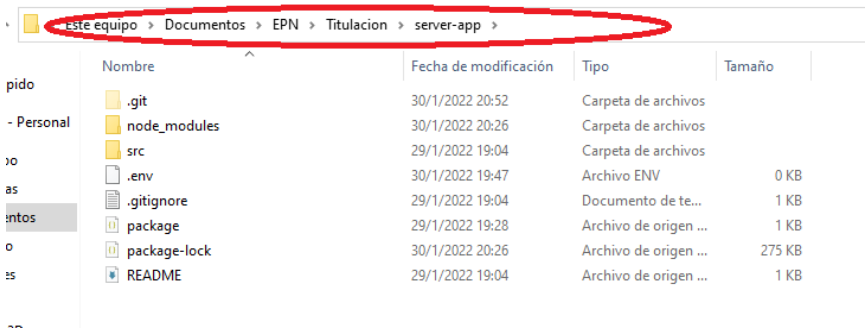
Ahora es necesario desplegar el back-end, por lo que se debe crear una cuenta de heroku donde se desplegara el servicio.

Para desplegar primero se necesita acceso a una computadora y descargar las herramientas de la documentación de la página.

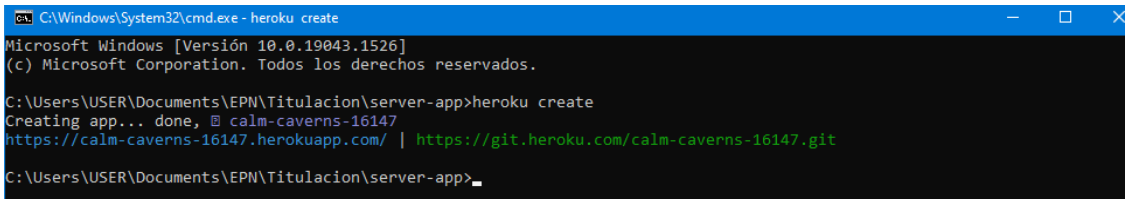


Ahora en una terminal del sistema operativo es necesario ingresar la cuenta de heroku con el comando (heroku login). Con ello el equipo donde se esté manejando.

En una consola de comandos a parte se debe ubicar en la carpeta del proyecto, se puede escribir cmd en el buscador de archivos y se abrirá el cmd en la ruta del proyecto.

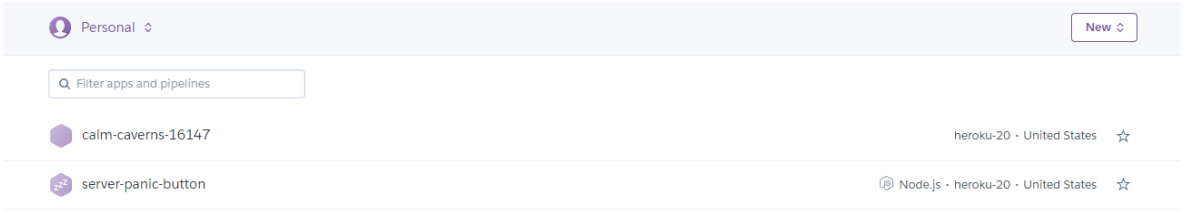


Ahora en la terminal se crea un proyecto de heroku con (heroku create).

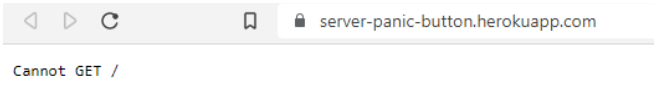


Y para poder desplegar el servicio a la nube se usa el comando (git push heroku master),

Si todo se realizó correctamente en la cuenta de heroku se podrá ver el proyecto creado y desplegado.



Al abrir el proyecto con el comando (heroku open), se debe visualizar esta pantalla.

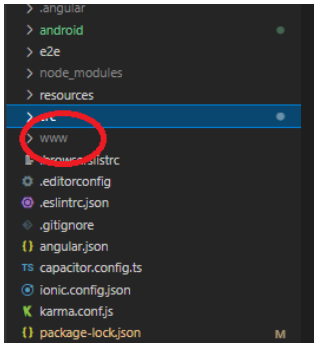


Creación de APK

La única configuración que se debe realizar en el código del front-end es agregar la url obtenida anteriormente en el archivo, contacts.service.ts como se ve en la captura de abajo

Ahora de la misma forma que con el back-end se debe abrir una consola de comandos en la hubicacion del proyecto y posteriormente ejecutar el comando (ionic build).

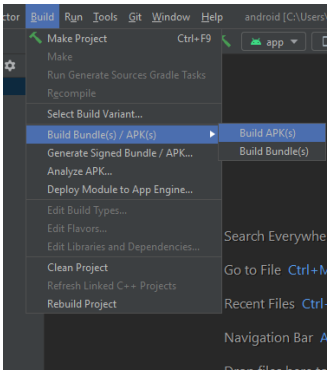
Esto crea el proyecto comprimido en una carpeta llamada www



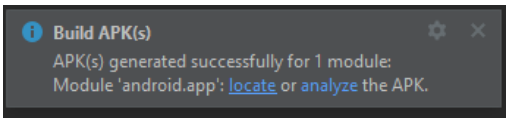
Ahora al ejecutar el comando (npx capacitor add Android) se crea el archivo que será transformado en el apk para la instalación en dispositivos móviles.

Una vez realizado, se ejecuta (npx capacitor open Android), para lo cual es necesario tener Android Studio en la computadora.

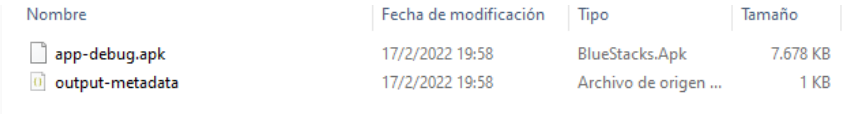
Cuando el proyecto se abra en el IDE de Android se debe generar el apk en la barra de herramientas que se muestran en la siguiente ilustración.



Cuando el proceso haya finalizado mostrara una alerta, en donde se debe dar clic en locate para dirigirse al apk.



El archivo llamado app-debug.apk es el que se debe instalar en el dispositivo móvil.



Al instalarlo en algunos dispositivos móviles, el propio sistema puede mandar una señal de alerta de seguridad, solo se debe dar permisos de instalación y la aplicación puede ser ya usada en el teléfono.

Nota: los enlaces para clonar los proyectos están en la bibliografía del documento principal.