UNIVERSIDAD UTE

FACULTAD DE CIENCIAS DE LA INGENIERÍA E INDUSTRIAS

CARRERA DE INGENIERÍA INFORMÁTICA Y CIENCIAS DE LA COMPUTACIÓN

Sistema visual ANDROID-Web de posicionamiento y guía en interiores mediante reconocimiento de marcado en el piso

TRABAJO PREVIO A LA OBTENCIÓN DEL TÍTULO

DE INGENIERO EN INFORMÁTICA Y CIENCIAS DE LA COMPUTACIÓN

RONNY OSCAR CALLE AJILA

DIRECTOR: ING. DIEGO ORDÓÑEZ

Quito, ABRIL 2022

© Universidad UTE. 2022
Reservados todos los derechos de reproducción

FORMULARIO DE REGISTRO BIBLIOGRÁFICO

TRABAJO DE TITULACIÓN

DATOS DE CONTACTO	
CÉDULA DE IDENTIDAD:	1722904982
APELLIDO Y NOMBRES:	CALLE RONNY OSCAR
DIRECCIÓN:	Eugenio Moreno N65-34 Y Libertador
EMAIL:	ronnyoscar_calleajila@hotmail.com
TELÉFONO FIJO:	02-2597637
TELÉFONO MÓVIL:	0999714224

DATOS DE LA OBRA	
TÍTULO:	Sistema visual ANDROID-Web de posicionamiento y guía en interiores mediante reconocimiento de marcado en el piso
AUTOR O AUTORES:	CALLE AJILA RONNY OSCAR
FECHA DE ENTREGA DEL PROYECTO DE TITULACIÓN:	Abril del 2022
DIRECTOR DEL PROYECTO DE TITULACIÓN:	ING. DIEGO ORDÓÑEZ
PROGRAMA	PREGRADO [X] POSGRADO [ ]
TÍTULO POR EL QUE OPTA:	INGENIERO EN INFORMÁTICA Y CIENCIAS DE LA COMPUTACIÓN
RESUMEN: Mínimo 250 palabras	El avance de tecnologías para el posicionamiento ha permitido tener una gran variedad de opciones para la implementación en el exterior que han resuelto la mayoría de los problemas que se tenían como la

	ubicación de un objeto o la navegación de un punto hacia otro. Sin embargo, cuando se habla de posicionamiento en interiores se ha mostrado la dificultad de poder utilizar las mismas tecnologías usadas para los exteriores, pero con el tiempo se ha ido innovando, desarrollando y aplicando nuevas tecnologías para mejorar el posicionamiento en interiores como el Beacon, W-Fi, RFID, INS. Por ello existe la necesidad de seguir buscando alternativas para lograr el objetivo de tener un sistema de posicionamiento en interiores. Para el desarrollo del sistema de posicionamiento se ha utilizado la metodología XP con la que se creó un aplicativo móvil Android y se diseñó un sistema de posicionamiento en interiores y direccionamiento, basado en el reconocimiento visual, discriminación y ubicación de señalización marcada en el piso, en donde se utilizó el código QR como la señalética de reconocimiento visual del marcado de piso y a través del aplicativo móvil se obtiene la información de posicionamiento en interior utilizando una base de datos en la nube que contiene información como latitud y longitud del punto QR en el

	establecimiento y mediante esto se pudo mostrar al usuario su posición actual y guiarlo hacia otro punto destino mediante el uso del algoritmo Dijkstra que permitirá mostrar la ruta corta entre 2 puntos, además de crearse una página Web para la creación, actualización y eliminación de puntos en mapa interior.
PALABRAS CLAVES:	QR, IPS, Indoor, Android, JavaScript
ABSTRACT:	The advancement of technologies for the positioning has made it possible to have a wide variety of options for implementation outside that have solved most of the problems that were encountered, such as locating an object or navigating from one point to another. However, when talking about indoor positioning, the difficulty of being able to use the same technologies used outdoors has been shown, but over time new technologies have been innovating, developing and applying new technologies to improve indoor positioning such as the Beacon, Wi-Fi, RFID, INS. Therefore, there is a need to continue looking for alternatives to achieve the objective of having an indoor positioning system. For the development of the positioning system, the XP methodology has been used with an

	Android mobile application and an indoor positioning and addressing system was designed. It based on visual recognition, discrimination and location of signaling marked on the floor, where the QR code was used as the visual recognition signage of the floor marking and through the mobile application. The indoor positioning information is obtained using a database in the cloud that contains information such as latitude and longitude of the QR point in establishment and through this it was possible to show the user his current position and guide him to another destination point through the use of the Dijkstra algorithm that will allow showing the short route between 2 points in addition to developing of a Web page for the creation, update and elimination of points on inside map.
KEYWORDS	QR, IPS, Indoor, Android, JavaScript

Se autoriza la publicación de este Proyecto de Titulación en el Repositorio Digital de la Institución.

_____

CALLE AJILA RONNY OSCAR

1722904982

DECLARACIÓN Y AUTORIZACIÓN

Yo, **CALLE AJILA RONNY OSCAR,** CI **1722904982** autor del trabajo de titulación: **Sistema visual Android-Web de posicionamiento y guía en interiores mediante reconocimiento de marcado en el piso** previo a la obtención del título de **INGENIERO EN INFORMÁTICA Y CIENCIAS DE LA COMPUTACIÓN** en la Universidad UTE.

1. Declaro tener pleno conocimiento de la obligación que tienen las Instituciones de Educación Superior, de conformidad con el Artículo 144 de la Ley Orgánica de Educación Superior, de entregar a la SENESCYT en formato digital una copia del referido trabajo de titulación de grado para que sea integrado al Sistema Nacional de información de la Educación Superior del Ecuador para su difusión pública respetando los derechos de autor.
2. Autorizo a la BIBLIOTECA de la Universidad UTE a tener una copia del referido trabajo de titulación de grado con el propósito de generar un Repositorio que democratice la información, respetando las políticas de propiedad intelectual vigentes.

Quito, abril del 2022

_____

CALLE AJILA RONNY OSCAR
1722904982

DECLARACIÓN JURAMENTADA DEL AUTOR

Yo, **CALLE AJILA RONNY OSCAR**, portador de la cédula de identidad N° **1722904982**, declaro que el trabajo 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.

La Universidad UTE puede hacer uso de los derechos correspondientes a este trabajo, según los establecido por la Ley de Propiedad Intelectual, por su reglamento y por la normativa institucional vigente.

_____

CALLE AJILA RONNY OSCAR
1722904982

CERTIFICACIÓN DEL TUTOR

En mi calidad de tutor, certifico que el presente trabajo de titulación que lleva por título **Sistema visual Android-Web de posicionamiento y guía en interiores mediante reconocimiento de marcado en el piso** para aspirar al título de **INGENIERÍA INFORMÁTICA Y CIENCIAS DE LA COMPUTACIÓN** fue desarrollado por **CALLE AJILA RONNY OSCAR,** bajo mi dirección y supervisión, en la Facultad de Ciencias de la Ingeniería e Industrias; y que dicho trabajo cumple con las condiciones requeridas para ser sometido a las evaluación respectiva de acuerdo a la normativa interna de la Universidad UTE.

ING. DIEGO ORDÓÑEZ

DIRECTOR DEL TRABAJO

C.I. 1710449016

AGRADECIMIENTOS

Primero agradezco a mi madre por todo el esfuerzo y sacrificio que ha dado día a día, por las enseñanzas, por los regaños, por los días bueno y malos que formaron a la persona que soy ahora, son varias las palabras que expresarían el cariño y aprecio hacia ella, pero la vida me ha dado la oportunidad de poder expresarlo con logros que no solo me enorgullecen a mi sino a ella.

Agradezco a mis hermanos, a mi prima y familia en general, por su apoyo, por insistir en que llegue alcanzar mis objetivos y por los consejos para seguir de pie día a día y por hacer de sus historias parte de mi aprendizaje,

Agradezco a mis compañeros que he logrado conocer durante toda mi carrera universitaria en especial a Patricio, Fabricio, Sebastián, Javier quienes han sido con los que más he compartido y con los que he logrado he logrado alcanzar metas, objetivos personales y grupales, de los cuales estoy seguro de que serán grandes profesionales.

Agradezco a mis profesores quienes desde el comienzo me han dado su apoyo y han sabido compartir su sabiduría, conocimiento y experiencia para permitirnos llegar a lograr nuestro objetivo de ser los nuevos profesionales y representantes de la carrera.

Agradezco al Ing. Diego Ordoñez, quien ha sido guía y apoyo para culminar el presente trabajo de titulación y quien ha sido ejemplo de cómo se debe ejercer una profesión.

Finalmente agradezco a quien estuvo durante toda mi carrera universitaria Candela Calle, que siempre estuvo a mi lado, que siempre me prestó atención, que se desvelaba a mi lado al culminar algún trabajo o proyecto, a quien logró que creciera en lo profesional y personal, a quien permitió que lograra disfrutar la vida y que lograra salir de cada situación en la que me sentí caído.

.

ÍNDICE DE CONTENIDO

PÁGINA

ÍNDICE DE FIGURAS

PÁGINA

ÍNDICE DE TABLAS

PÁGINA

ÍNDICE DE ANEXOS

PÁGINA

RESUMEN

El avance de tecnologías para el posicionamiento ha permitido tener una gran variedad de opciones para la implementación en el exterior que han resuelto la mayoría de los problemas que se tenían como la ubicación de un objeto o la navegación de un punto hacia otro. Sin embargo, cuando se habla de posicionamiento en interiores se ha mostrado la dificultad de poder utilizar las mismas tecnologías usadas para los exteriores, pero con el tiempo se ha ido innovando, desarrollando y aplicando nuevas tecnologías para mejorar el posicionamiento en interiores como el Beacon, W-Fi, RFID, INS. Por ello existe la necesidad de seguir buscando alternativas para lograr el objetivo de tener un sistema de posicionamiento en interiores. Para el desarrollo del sistema de posicionamiento se ha utilizado la metodología XP con la que se creó un aplicativo móvil Android y se diseñó un sistema de posicionamiento en interiores y direccionamiento, basado en el reconocimiento visual, discriminación y ubicación de señalización marcada en el piso, en donde se utilizó el código QR como la señalética de reconocimiento visual del marcado de piso y a través del aplicativo móvil se obtiene la información de posicionamiento en interior utilizando una base de datos en la nube que contiene información como latitud y longitud del punto QR en el establecimiento y mediante esto se pudo mostrar al usuario su posición actual y guiarlo hacia otro punto destino mediante el uso del algoritmo Dijkstra que permitirá mostrar la ruta corta entre 2 puntos, además de crearse una página Web para la creación, actualización y eliminación de puntos en mapa interior.

Palabras Clave: QR, IPS, Indoor, Android, JavaScript

ABSTRACT

The advancement of technologies for the positioning has made it possible to have a wide variety of options for implementation outside that have solved most of the problems that were encountered, such as locating an object or navigating from one point to another. However, when talking about indoor positioning, the difficulty of being able to use the same technologies used outdoors has been shown, but over time new technologies have been innovating, developing and applying new technologies to improve indoor positioning such as the Beacon, Wi-Fi, RFID, INS. Therefore, there is a need to continue looking for alternatives to achieve the objective of having an indoor positioning system. For the development of the positioning system, the XP methodology has been used with an Android mobile application and an indoor positioning and addressing system was designed. It based on visual recognition, discrimination and location of signaling marked on the floor, where the QR code was used as the visual recognition signage of the floor marking and through the mobile application. The indoor positioning information is obtained using a database in the cloud that contains information such as latitude and longitude of the QR point in establishment and through this it was possible to show the user his current position and guide him to another destination point through the use of the Dijkstra algorithm that will allow showing the short route between 2 points in addition to developing of a Web page for the creation, update and elimination of points on inside map.

Keywords: QR, IPS, Indoor, Android, JavaScript

1. INTRODUCCIÓN

1. INTRODUCCIÓN

El uso de mapas para movilización en exteriores ha sido uno de los desarrollos más importantes de la tecnología. Sin embargo, cuando se habla de mapas de interiores las cosas cambian debido a que la tecnología que se tiene no trabaja de manera adecuada dentro de un mapa de interior, como ejemplo se tiene aplicaciones que permiten movilizarnos de un lugar a otro y mostrarnos nuestra ubicación actual a través del uso del GPS y sensores incluidos en los dispositivos que se han ido optimizando para trabajar de manera eficaz para exteriores.

En cambio, una persona que se encuentre dentro de un establecimiento tiene como problemas que no tiene la suficiente información de su ubicación y la dirección a donde dirigirse para llegar a su destino, debido a que se usa tecnologías basadas para la ubicación de exteriores y que no permite mostrar el posicionamiento de la persona de una manera eficaz para interiores.

Otro problema para el posicionamiento en interiores son los mapas de geolocalización, ya que están planeados para trabajar en entornos exteriores, por lo tanto, no se tiene información de mapas internos y las instituciones privadas o públicas tienen información que no podrá mostrarse por seguridad.

En (Zhuang et al., 2018), se realizó un framework para navegación en interiores basado en código QR, una de las ventajas que menciona en la implementación para la navegación en interiores con respecto a otras tecnologías es la facilidad de obtener información del código QR sin necesidad de tener una conexión a internet y poder trabajar con una base de datos.

En(Mamaeva et al., 2019), realizaron una aplicación que trabaja con código QR como identificador de la posición de un nodo dentro de mapa de interiores, y para la navegación utilizan un algoritmo de ruta corta junto a la realidad virtual para guiar al usuario dentro un mapa interior, y se muestra como ventaja el bajo costo de implementación con respecto a otras tecnologías de posicionamiento en interiores.

En (Rantanen et al., 2018),se describe un sistema sin infraestructura es decir donde no existen elementos preinstalados como el Bluetooth, Wi-Fi ni conocimientos preliminares como mapas o planos del lugar, sino que utilizan el sistema de navegación inercial (INS) con el objetivo de que se pueda utilizar en cualquier lugar y sostiene que la limitación que tiene este sistema es la estimación relativa de posición.

En (Aguilar Herrera et al., 2014) describen la importancia de la ubicación en interiores por parte de los peatones, y plantean el uso de 2 sistemas que son el de navegación inercial y el uso de balizas (beacons) junto al algoritmo PDR que se trata de una estimación de la trayectoria del peatón mediante los sensores como acelerómetro y giroscopio, además se incluye un mapa de interiores elaborado en OpenStreetMap para tener la mayor información de la infraestructura donde se quiere utilizar este sistema.

En (Cankaya et al., 2015), se realizó un sistema de navegación de interiores usando la realidad virtual desde un dispositivo móvil, una de las ventajas de este tipo de sistema es que a través de la realidad virtual existe la posibilidad del reconocimiento de objetos con lo cual se puede determinar una posición de la persona dentro del interior y que la información se muestra directamente desde el dispositivo de manera interactiva.

En la actualidad existen diferentes tipos de tecnología que permiten mostrar la ubicación de una persona,

GPS (Global Positioning System). Se trata de un sistema diseñado por el departamento de defensa de los Estados Unidos para ser usado por los militares con el objetivo de obtener la información de la posición, localidad y tiempo (Pozo Ruz et al., 2000). Para lograr ese objetivo se hacía uso de los satélites los cuales mediante una triangulación muestran como información la altitud, longitud y latitud.

IPS (Indoor Positioning System). Se trata de la detección de la ubicación de un objeto mediante distintas técnicas y algoritmos, en donde los datos pueden estar en forma de la intensidad o el alcance de la señal(Batistić & Tomić, 2018). A continuación, se describirán algunos métodos y algoritmos que se utilizan para estimación de la posición de un objeto.

Triangulación. Se trata de utilizar las propiedades de los triángulos para determinar la distancia o la orientación.

Lateración. Se puede utilizar de dos maneras, la primera se basa en el tiempo que tarda una señal en viajar desde un transmisor a un receptor, para calcular la distancia entre los 2 puntos denominada TOA (Time of arrival), la Figura 1 muestra el principio de triangulación del TOA mediante el cual determina la distancia del punto necesitado con los receptores.

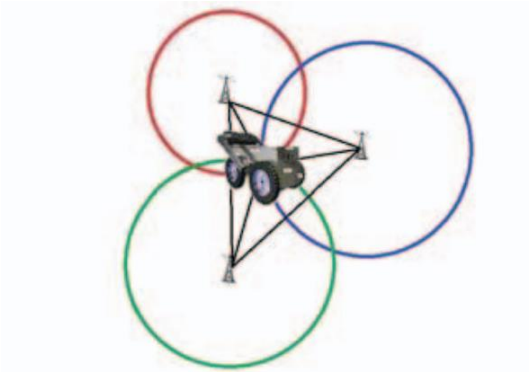


Figura 1. *Principio de la triangulación TOA para determinar la ubicación* (Rashid et al., 2016).

En la ecuación [1] se tiene la fórmula de una ecuación lineal simple, donde c es la velocidad de la Luz, t el tiempo (Batistić & Tomić, 2018)

$$d = c * t \qquad [1]$$

Para un espacio en 2D se tiene que en la ecuación [2] cambia y se agregan (x_{ref}, y_{ref}) que son coordenadas de una referencia conocida y los puntos (x, y) que son las coordenadas de un objetivo (Batistić & Tomić, 2018)

$$d = \sqrt{(x_{ref} - x)^2 + (y_{ref} - y)^2} \quad [2]$$

Mediante las ecuaciones [1] y [2] se calcula la posición del punto objetivo, sin embargo, para ello se necesita de al menos tres puntos de referencia para determinar la ubicación y como se basa en la en el tiempo de llegada de la señal, los relojes de los dispositivos deben estar sincronizados.

La segunda se denomina TDOA (Time difference of arrival) tiene como característica el determinar la diferencia de tiempo en que llega la misma señal a múltiples receptores (García et al., 2015).

En la Figura 2 se visualiza la diferencia que el método TOA manda la señal a un solo receptor para posteriormente medir la distancia entre esos 2 puntos mientras que el método TDOA el dispositivo manda la misma señal a varios receptores.

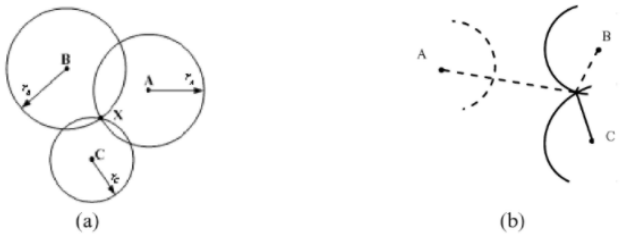


Figura 2. Boceto de un TOA (a) y TDOA (b) (Shen & Huang, 2011).

Angulación. Se la denomina DOA (Direction of arrival), y determina la ubicación del objetivo estimando la intersección de direcciones a las que llegan las señales de recepción, y para ello se necesita de dos ángulos.

Fingerprinting. Está basada en dejar un dispositivo con huella dactilar en este caso por radio. Para trabajar con ella se tiene un base de datos la cual mediante RSSI (Received signal strength indication) medido del objeto se compara con el referido en busca del más cercano(Subedi et al., 2019).

Modelado de propagación de señales. Se utiliza modelos matemáticos de propagación de señales en la cual se tiene un conjunto de datos de intensidad de señal realizado y un conjunto de datos calculado teóricamente, que posteriormente para encontrar la ubicación del usuario se estima por la coincidencia de la señal medida en tiempo real y la señal medida teóricamente(Batistić & Tomić, 2018).

Proximidad. Esta técnica de localización por proximidad proporciona información sobre la ubicación de un objetivo con respecto a una posición conocida en una zona. La ubicación que se proporciona es un relativo aproximado ya que cuando el dispositivo sea detectado en un punto de referencia, se considera que está cerca de ese punto de referencia(Batistić & Tomić, 2018).

A continuación, se describirán algunas tecnologías usadas para la ubicación en interiores.

Wireless Local Area Network. Para la utilización de esta tecnología se hace uso de las técnicas de Fingerprinting y la propagación de señales. Para lo cual se mide la intensidad de la señal para cada punto de acceso y modelado de la propagación de la señal(Batistić & Tomić, 2018). En el sistema de posicionamiento Ekahau utiliza una infraestructura WLAN existente, este sistema consta de puntos de acceso, etiquetas, software de mapeo y calibración, en donde las etiquetas sirven para los usuarios o los objetos que van transmitiendo una señal de Wi-Fi periódicamente(Li et al., 2013).

Identificación por radiofrecuencia (RFID). Se utilizan ondas de radio para almacenar y recuperar datos entre un lector y una etiqueta, Los lectores de RFID son capaces de leer los datos emitidos desde etiquetas RFID. La ventaja de esta tecnología es su bajo costo y flexibilidad para un uso general en la identificación de objetos(Batistić & Tomić, 2018).

Existen 2 tipos de tecnología RFID que son: activos y pasivos.

El RFID pasivo se trata de que la etiqueta rastreada es un receptor, la etiqueta sirve para modular y reflejar la señal al trasmisor sin necesidad de una fuente de energía que permite que su precio sea relativamente bajo(Wang & Pan, 2012).

El RFID activo se trata de que el dispositivo rastreado es un trasmisor, este dispositivo actúa como una baliza y transmite periódicamente señales que contiene la identificación y tiene como desventaja que necesita de energía para transmitir una señal incrementando el costo de implementación(Wang & Pan, 2012).

Ultrasonidos. Para la tecnología de posicionamiento por ultrasonidos se utilizan frecuencias sonoras que están por encima del límite superior audible de una persona. Para determinar la posición del objetivo se mide el tiempo necesario para que una señal viaje de un transmisor a un receptor(Díaz et al., 2017).

En este tipo de tecnología se utiliza la técnica de triangulación mediante la medición de señales TOA, por lo tanto, al menos 3 receptores necesitan recibir la señal del objetivo.

Sistemas de navegación de inercia. Estos permiten utilizar los sensores del dispositivo para medir la distancia recorrida, orientación de movimiento del dispositivo y con esto se puede obtener el movimiento del usuario. La gran ventaja a comparación de otras tecnologías es que no necesita referencias externas para obtener la información de movimiento del usuario(Descamps-Vila et al., 2013).

Las metodologías ágiles se diferencian de las tradicionales en los procesos, en el contexto y organización. Para ello en la Tabla 1 se enumera las principales diferencias de las metodologías ágiles y tradicionales en donde se interpreta que un proyecto con pocas personas, donde exista flexibilidad de contrato, donde no se tiene procesos controlados y se basa en la heurística de prácticas de producción de código, lo más recomendable es la utilización de una metodología ágil.

Tabla 1: *Diferencias entre metodologías ágiles y metodologías tradicionales.*

Metodologías Ágiles	Metodologías Tradicionales
Basadas en heurísticas provenientes de prácticas de producción de código	Basadas en normas provenientes de estándares seguidos por el entorno de desarrollo
Especialmente preparados para cambios durante el proyecto	Cierta resistencia a los cambios
Impuestas internamente (por el equipo de desarrollo)	Impuesta externamente
Proceso menos controlado, con pocos principios	Proceso mucho más controlado, con numerosas políticas/normas
No existe contrato tradicional o al menos es bastante flexible	Existe un contrato prefijado
El cliente es parte del equipo de desarrollo	El cliente interactúa con el equipo de desarrollo mediante reuniones
Grupos pequeños (<10 integrantes) y trabajando en el mismo sitio	Grupos grandes y posiblemente distribuidos
Pocos roles	Más roles
Menos énfasis en la arquitectura del software	La arquitectura del software es esencial y se expresa mediante modelos

(Letelier & Penadés, 2012)

A continuación, se presentan los conceptos sobre el lenguaje de programación, gestor de base de datos, librerías, herramientas y algoritmos utilizados para cumplir con el objetivo del proyecto.

Códigos QR. Es un código de barras que se representa de manera gráfica en el que se puede almacenar información que sirva como identificador con respuestas rápidas, la información que se puede almacenar es tipo texto que pueden llegar a hacer como SMS, URL, geolocalización, Email, Wi-Fi. En la Figura 3 se visualiza los 2 tipos de código de barras que se pueden utilizar

como es el unidimensional (A) y el que representaría el QR que se trata de un código de barras bidimensional (B)(Chaudhery, 2021).

Figura 3. Código de barras unidimensional (A) y bidimensional(Chaudhery, 2021).

Algoritmo de Dijkstra. Es uno de los algoritmos de ruta corta más populares, que fue formulado por el científico Edsger Dijkstra en el año 1956. En el algoritmo de Dijkstra trata de encontrar la ruta más corta de un gráfico con costos de ruta que no sea negativos, para ello tiene que procesar cada nodo del gráfico una vez y por lo tanto el número de nodos a evaluar determina la velocidad del algoritmo para encontrar la ruta más corta. (Candra et al., 2020).

En la ecuación [3] se muestra la complejidad del algoritmo de Dijkstra en un grafo con una matriz de adyacencia o con una lista de adyacencia en donde N representa el número de vértices, y en cada bucle es N-1 que va disminuyendo en cada interacción(Basu, 2005).

$$O(N^2) \qquad [3]$$

Base de datos NoSQL. La aparición del BIG DATA tiene como implicación la persistencia de la información, durante mucho tiempo se ha utilizado la base de datos relacionales. En (Sarasa, 2019) se define características de la base de datos NoSQL como: la mayoría nacen de proyectos de código abierto, el lenguaje para la recuperación de la información no es SQL, la mayoría está diseñada para ejecutarse en ambientes distribuidos basado en clústeres de máquinas y no tienen un esquema fijo.

Además de ello señala algunos tipos de base de datos NoSQL que se describirán a continuación:

Base de datos clave-valor. Son un tipo de base de datos orientada hacia agregados que puede llegar hacer de cualquier tipo que utiliza un método simple de clave-valor para almacenar lo datos; ofrece una gran velocidad de búsqueda y fácil de escalar pero que no puede emplear métodos complejos de consulta como en una base de datos SQL (Joyanes Aguilar, 2016).

Base de datos basada en columnas. Se trata de una base de datos que distribuye los datos en columnas, se la utiliza para cuando se necesita analizar grandes cantidades de datos a comparación de una base de datos en filas que permite trabajar en aplicaciones de transacciones (Sarasa, 2019).

Firebase Realtime Database. Es una base de datos alojada en la nube que permite el almacenamiento de los datos en un formato JSON y que permite sincronizar en tiempo real en cada dispositivo conectado. (Firebase, 2022).

A continuación, se describe las siguientes funciones claves dentro de Firebase Realtime Database (Firebase, 2022):

Tiempo Real. Permite hacer una sincronización de datos entre todos los dispositivos conectados y permitiendo que los datos dentro de un dispositivo permanezcan siempre actualizados.

Sin Conexión. Permite que los datos persistan dentro del dispositivo hasta que se restablezca la conexión y pueda nuevamente sincronizar con el servidor.

Acceso de distintos dispositivos clientes. Permite el acceso desde dispositivos móviles o un navegador web, además de proporcionar seguridad y validaciones de datos a través de reglas de seguridad.

Escalabilidad. Permite trabajar con varias instancias de base de datos dentro de un mismo proyecto, la autenticación de cliente en varias instancias de la base de datos.

Android. En el año 2008, se anunció la publicación del código fuente de la plataforma Android en open source. Donde se podía descargar, compilar, ejecutar y contribuir en su evolución, además que incluye el Android Market para que los desarrolladores pudieran poner sus aplicaciones para los clientes (Hébuterne, 2016). En las últimas versiones lanzadas se dispone de mejoras enfocadas a los desarrolladores que permiten una mejor optimización de las aplicaciones con respecto a la industria del 5G, la grabación de pantalla integrada en el sistema.

Finalmente, en la última versión Android 12 lanzada en la actualidad tiene un nuevo rediseño de la interfaz de usuario del sistema a través del nuevo lenguaje visual denominado Material You, también se agregan mejoras con respecto a la privacidad del usuario con lo cual notifica de las apps que han accedido algún elemento como la cámara, el micrófono y la ubicación en las últimas horas.

Java. Es un lenguaje de programación que se deriva de C y C++, sus aplicaciones son compiladas a bytecode que permite ejecutar en cualquier máquina virtual Java (MVJ) sin importar la arquitectura del ordenador (Baesens, Backiel, & vanden Broucke, 2015).

El lenguaje de programación Java permite realizar aplicaciones web, servicios web basado en SOAP o REST, aplicaciones de escritorio de consola o interfaz gráfica y además es el lenguaje de programación que se usa para el desarrollo

nativo para Android, lo cual ha permitido que este lenguaje sea uno de los más usados por la alta demanda de profesionales, sin embargo, también existen otros lenguajes como lo son Dart y Flutter.

JavaScript. Es un lenguaje de programación que funciona en los navegadores de forma nativa permitiendo tener una mejor interactividad y dinamismo a una página web. En sus inicios JavaScript solo se utilizaba desde el lado del cliente, pero al pasar los años se ha creado nuevas versiones que permiten ser ejecutado desde el servidor como Node JS, con lo cual se puede utilizar tanto para el Back-end y Front-end (Flanagan, 2011).

Node JS. Es un entorno en tiempo de ejecución multiplataforma para la capa del servidor basado en JavaScript, que tiene una arquitectura basada en eventos mediante un único hilo que ejecuta el código de la aplicación y utiliza un modelo de E/S asíncrona, que permite trabajar en aplicaciones intensivas de CPU en tiempo real que se ejecutan en muchos dispositivos y que a su vez está diseñado para crear aplicaciones escalables permitiendo establecer y gestionar múltiples conexiones al mismo tiempo(Chitra & Satapathy, 2017).

Express.js. Es un marco de aplicaciones web gratuito para Node.JS, el cual se utiliza para diseñar y construir aplicaciones Web de forma rápida y sencilla. Express.js es ligero y ayuda a organizar las aplicaciones web en el lado del servidor en una arquitectura MVC. Express.js tiene como características el desarrollo rápido del servidor, enrutamientos que ayudan a preservar el estado de la página Web con la ayuda de los URL.

Handlebars. Es un sistema de plantillas en JavaScript que permite crear y formatear código HTML de una manera sencilla, mediante esta plantilla se puede definir la estructura que se presentará la información, además de reutilizar secciones de código. Básicamente el uso de una plantilla permite separar, optimizar y organizar el código de desarrollo para una mejor eficiencia y productividad (Manricks, 2013).

Mapbox. Es una plataforma de mapeo y ubicación en la nube que incluye una gran cantidad de productos y servicios para los desarrolladores, en donde pueden hacer diseños de mapas personalizados y creación de aplicaciones (mapbox, 2022).

Maps. Es un producto que proporciona herramientas de diseño y de bibliotecas necesarias para la creación de mapas dinámicos, eficaces y personalizados que se adapten a los requerimientos de desarrollo de un sistema (Climent, 2019).

Studio. Es un producto de Mapbox que permite tener el control completo para el diseño y personalización de mapas, que posteriormente se puede publicar y añadir a una aplicación web o móvil(Kastanakis, 2016).

QGIS. Es un sistema de información geográfica de código abierto, este proyecto nació en mayo del 2002 con la intención de ofrecer los mismos servicios del software GIS que es un software propietario, pero con la diferencia de que esté disponible para cualquier persona, ya que está basado en una licencia GNU (General Public License). Este software está soportado para la mayoría de las plataformas Unix, Windows y macOS. QGIS se desarrolla utilizando el kit de herramientas Qt y C++ (QGIS project, 2022).

El presente trabajo de titulación tiene como objetivo general implementar un sistema de posicionamiento en interiores y direccionamiento, basado en el reconocimiento visual, discriminación y ubicación de señalización marcada en el piso. Se definen los siguientes objetivos específicos: concebir un sistema integrado de señalética marcada en el piso, que permita su reconocimiento para la ubicación de un usuario en un espacio interior; implementar una aplicación Web de administración, configuración y uso de sistema de señalización; desarrollar una aplicación Android para presentar al usuario la ubicación actualizada del dispositivo, y el camino a seguir basada en la detección visual de la señalización.

2. METODOLOGÍA

2. METODOLOGÍA

Para el desarrollo del trabajo de investigación se tomó como base las metodologías ágiles específicamente la XP “Extreme Programming”. Esta metodología está enfocada en potenciar las relaciones interpersonales, donde se promueve la comunicación y el trabajo en equipo, además de preocuparse por el aprendizaje del desarrollador(Letelier & Penadés, 2012).

Se debe familiarizar con las herramientas y tecnologías que se emplearán en el proyecto, en la Tabla 2 se muestran las herramientas de hardware para la realización del proyecto.

Tabla 2. Herramientas y tecnologías empleadas en el proyecto**.**

Herramienta	Descripción
Windows 10 Profesional 64-bits	Sistema operativo donde fue desarrollado la aplicación
Android Studio Arctic Fox 2020.3.1	IDE donde fue desarrollado el aplicativo Android
Mapbox	APIs y SDK utilizado para visualizar el mapa de interiores
Firebase	APIs y SDK utilizado para el almacenamiento de datos NoSQL
QGIS	Software para el georreferenciado de nuestro mapa interior
Visual Studio Code 1.63.2	Editor de código para la página Web
Node.js, express.js	Servidor para la página web
Dell Laptop Inspiron 14 7000, Intel Core i7-8550U 1.80GHz 1.99 GHz, 8 GB RAM	Equipo donde se desarrolló el aplicativo Android y la página Web
Huawei P20 Lite (Android 9)	Dispositivo móvil que se probó el aplicativo Android.

Extreme Programming está compuesta por seis fases las cuales son: exploración, planificación de entrega, Iteraciones, producción, mantenimiento, muerte del proyecto. Sin embargo, para el desarrollo de un software se definen 4 procesos que son: planeación, diseño, desarrollo y pruebas.

2.1 PLANEACIÓN

En esta fase se procedió a definir la aplicación que se deberá entregar y con cada iteración se deberá obtener un software útil, funcional y listo para probar y lanzar.

Para la definición de la aplicación se recolectó información de los requerimientos del cliente como historias de usuario permitiendo definir los requerimientos funcionales y los módulos a realizarse para la aplicación.

En la Tabla 3 se definen las actividades realizadas para esta fase

Tabla 3. Tabla de actividades para la fase de planeación.

Actividades	Encargado
Identificación de requerimientos	Ronny Calle
Identificación de módulos	Ronny Calle

2.2 DISEÑO

Se establece la prioridad de los requerimientos establecidos en la fase de planeación y con ello se realiza una estimación del esfuerzo para cada una de ellas. Se acuerda el contenido de entrega y cronograma que no podrá llevar más de tres meses. En la Tabla 4 se visualiza las actividades a realizarse con respecto a los requerimientos.

Tabla 4. Actividades de la fase de diseño.

Actividades	Descripción	Encargado
Diseño del mapa de interiores	Se diseñó el mapa de interiores que será utilizado para el aplicativo móvil y web mediante Mapbox	Ronny Calle
Diseño de la base de datos	Se estableció un modelo de base de datos NoSQL para gestionar los marcadores que se utilizaran en el aplicativo móvil	Ronny Calle
Diseño de la interfaz gráfica del aplicativo móvil y web	Se estableció un diseño funcional y amigable para el usuario	Ronny Calle

2.3 DESARROLLO

En esta fase ya se desarrolla los puntos de cada iteración con sus respectivos programadores y al finalizar muestra al cliente la iteración lograda, donde se hacen además pruebas adicionales y de rendimiento. En la Figura 4 se muestra el cronograma que se realizaría de las iteraciones cuando recién se inició el proyecto el cual por cada iteración puede alterar el cronograma debido a nuevas características que el cliente solicite para el aplicativo Android o página Web.

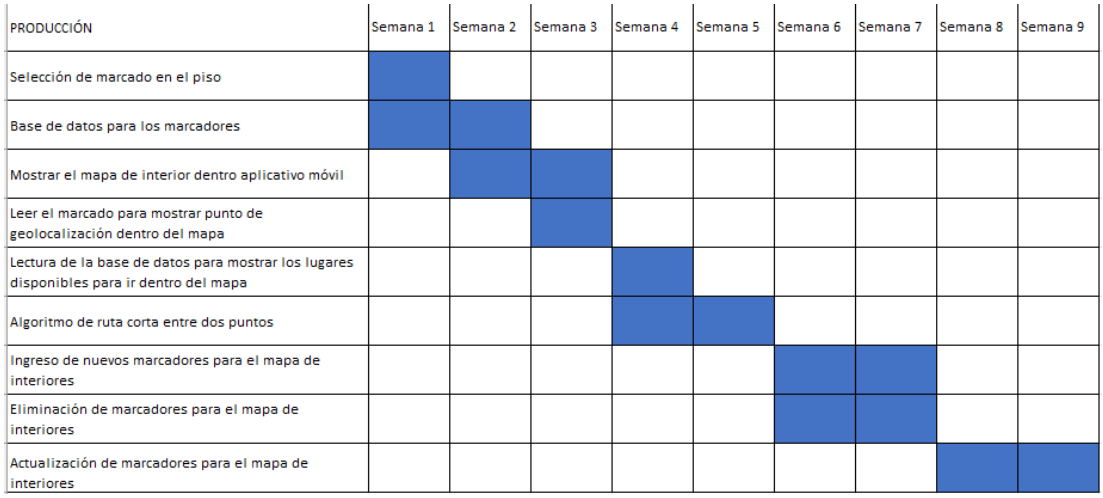


Figura 4. Cronograma del desarrollo de iteraciones.

2.3.1 Marcador en el piso

Se buscó definir qué tipo de marcado se podría usar como identificador para al aplicativo Android para mostrar la posición actual de la persona dentro del mapa de interiores. Para lograr este apartado se trató de definir qué tipo de marcado, señalética e información podría detectarse por medio del dispositivo móvil.

2.3.2 Visualización del mapa de interior

Se buscó herramientas que permitieran mostrar el mapa de interior en el dispositivo Android entre las opciones que se tuvo en cuenta fueron:

- Google Maps
- Mapbox
- OpenStreetMap

Cada una de ellas proporcionan características que pueden influir en la selección de la herramienta a usarse para la visualización del mapa de interiores, por ejemplo, tanto Mapbox y OpenStreetMap proporcionan servicios gratuitos con lo cual su costo de implementación puede llegar a ser menor que Google Maps. En Google Maps se puede tener un mapa de interiores que trabaje con la API o SDK de Google, pero para la implementación del mapa se requiere hacer la solicitud de diseño de un mapa de interior en el cual posiblemente se necesite un mapa más elaborado y que

llegue a cumplir con los requisitos que requiere Google Maps para su implementación.

Mapbox proporciona herramientas gratuitas que permite elaborar mapas sin ningún diseño e incorporar el mapa de interiores propios y hacer uso de nuevas capas que muestren leyendas precargadas de los establecimientos o lugares que se encuentre dentro del mapa del interior.

2.3.3 Base de datos

Para el almacenamiento se procedió a buscar una base de datos que permita comunicarse tanto con la página web y el aplicativo Android, para lo cual se tomaron en cuenta base de datos SQL y NoSQL. En la cual cada una proporciona ventajas y desventajas con respecto a su uso, costo, fiabilidad y rendimiento. Se buscó definir la mejor opción para usar como almacenamiento de datos donde como principales características tenga la escalabilidad, rendimiento y la vez que con el tiempo se puede mejorar y añadir nuevas características de uso con la página Web y el aplicativo Android.

Como principales opciones para la implementación de una base de datos son Firebase, MongoDB

2.3.4 Algoritmo ruta corta

Para que el aplicativo Android permita la opción de mostrar la ruta más corta entre dos puntos, se busca usar un algoritmo que maneje las conexiones entre los puntos y los pesos que tienen entre ellos.

2.3.5 Página Web

Se creó la página con el uso de las herramientas planteadas y de acuerdo con el diseño elaborado para la página Web en el que permita ingresar, actualizar, eliminar nuevos puntos que se serán almacenados en la base de datos para que luego trabajen dentro del aplicativo Android. Se consideró que servidor se debería utilizar para la implementación de la página Web la cual se realizó con el lenguaje de JavaScript y este lenguaje al poder utilizarse tanto como el Back-End y Front-End permite trabajar tanto el servidor y cliente con el mismo lenguaje

2.4 PRUEBAS

En esta fase se toman las decisiones de la inclusión de nuevas características a la versión actual para que puedan ser implementadas en las siguientes iteraciones. También se pueden hacer pruebas de desempeño de la iteración y de integración con otras iteraciones realizadas anteriormente.

3. RESULTADOS Y DISCUSIÓN

3. RESULTADOS Y DISCUSIÓN

El sistema realizado se basa en un aplicativo móvil que permite a través de un marcado en el piso visualizar su posición dentro del mapa de interiores y que además permite mostrar la ruta corta desde ese punto inicial al punto de llegada deseado y tambien se necesita de una aplicación web que permita crear, editar y eliminar marcadores que son utilizados desde el aplicativo móvil.

3.1 PLANEACIÓN

Se realizó una reunión para obtener la información necesaria que permitieron definir los requerimientos. Las historias de usuario se muestran en el Anexo 1. HISTORIAS DE USUARIO.

En la Tabla 5 se visualiza los procesos a realizarse, la prioridad e iteración en la que se realizará cada proceso.

Tabla 5. Tabla de procesos de prioridad e iteración.

N°	Nombre	Prioridad	Riesgo	Esfuerzo	Iteración
1	Selección de marcado en el piso	Alta	Medio	3	1
2	Mostrar el mapa de interior dentro aplicativo móvil	Alta	Medio	3	1
3	Leer el marcado para mostrar punto de geolocalización dentro del mapa	Media	Medio	2	1
4	Visualización de los marcadores en el aplicativo	Alta	Medio	3	2
5	Algoritmo de ruta corta entre dos puntos	Alta	Alto	4	2
6	Ingreso de nuevos marcadores para el mapa de interiores	Media	Medio	3	3
7	Eliminación de marcadores para el mapa de interiores	Baja	Medio	3	3
8	Actualización de marcadores para el mapa de interiores	Baja	Medio	3	3
9	Generar el código QR	Baja	Bajo	3	4

3.2 DISEÑO

En el diseño se realizó la creación de un mapa de interiores, una base de datos NoSQL y la interfaz gráfica del aplicativo móvil y web

3.2.1 Creación de estilo de mapa de interior dentro de Mapbox

Para crear un estilo del mapa de interiores se ha usó QGIS para hacer una georreferencia del mapa de interior sobre un mapa de Mapbox, Google o OpenStreetMap, el software de QGIS es open source y proporciona las mismas herramientas que GIS el cual es un software privado. En la Figura 5 se visualiza el software de QGIS y en donde se muestra el panel de navegación y el panel de capas del proyecto en donde deberá arrastrar desde el navegador al panel de capas el mapa que se desea visualizar para hacer la georreferencia del mapa de interiores y en este caso se utilizó el mapa de Google Maps el cual permite tener la opción de varios tipos de mapa como lo son el normal, satelital y rutas.

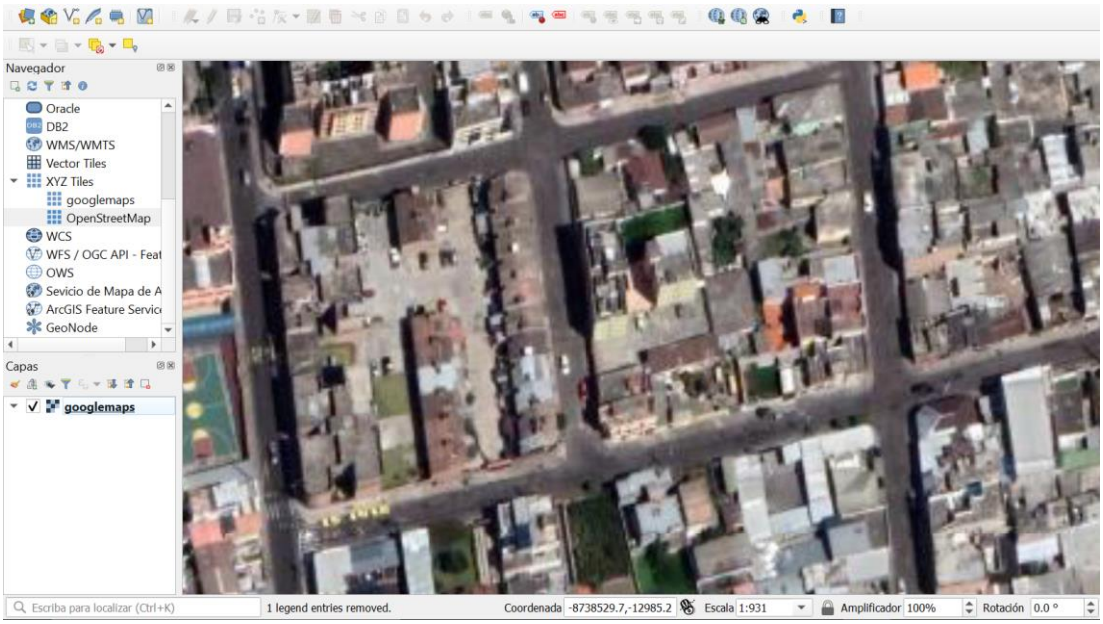


Figura 5. Herramienta QGIS añadiendo el mapa de Google Maps.

Dentro de QGIS existe la herramienta Georreferenciador la cual necesita de una imagen que puede ser tipo PNG, JPG. Para el mapa de interiores se utilizó una imagen tipo PNG, en la Figura 6 se muestra cómo se van ubicando los puntos de referencia desde la imagen hacia el mapa Google Maps y en la tabla PCT se muestra los puntos ya georreferenciados.

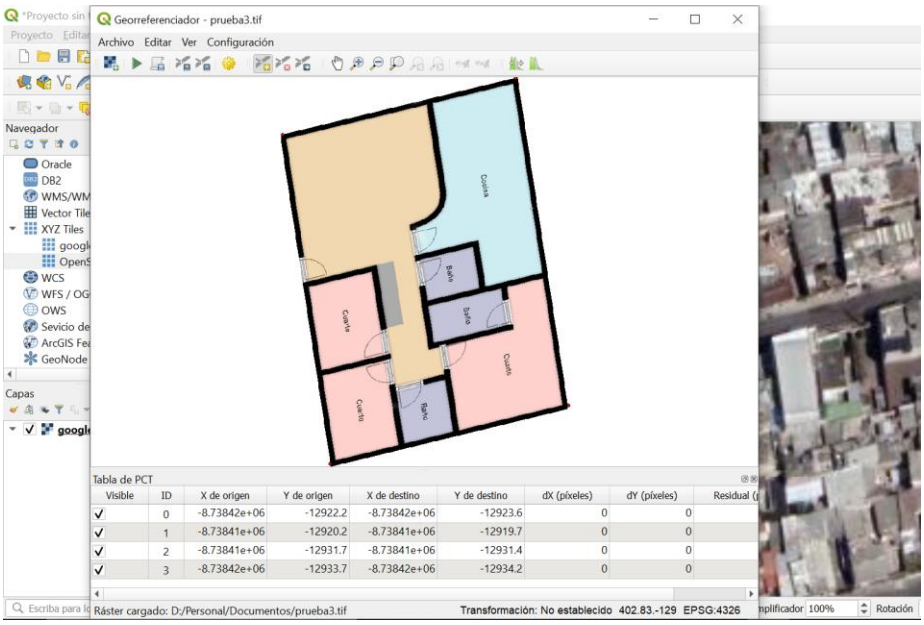


Figura 6. Realización del georreferenciado del mapa de interior con QGIS.

Para integrar la información de georreferenciado se ejecuta la herramienta y no es necesario cambiar ninguna de las opciones por defecto que se proporciona desde la herramienta y en la Figura 7 se muestra la configuración que se ejecutará (A) y como se visualizará (B) el mapa de interiores ya en un mapa como Google Maps, Mapbox, OpenStreetMap y otros.

(A) (B)

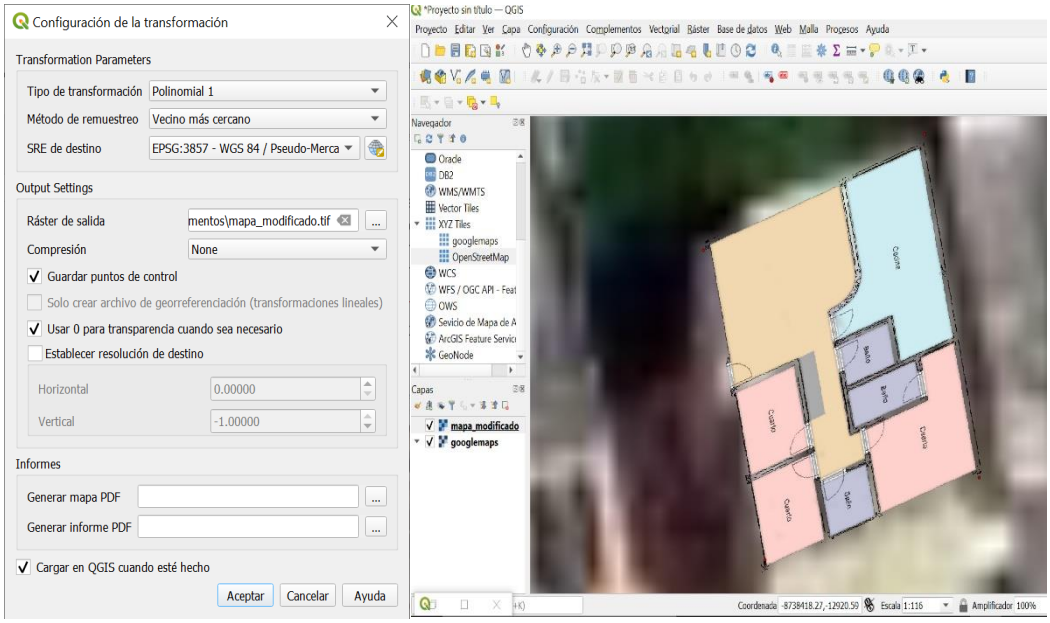


Figura 7. Configuración básica para el georreferenciado (A) y visualización del georreferenciado de mapa interiores (B).

Al realizar el georeferenciado se exportara un archivo .TIF que permitirá cargarlo dentro de Mapbox Studio. Mapbox Studio será la herramienta que permita crear un estilo de mapa que se integrará para el aplicativo móvil. Para ello dentro de Mapbox studio se crea un nuevo mapa con el estilo que se deseado para el mapa en el que se muestre las calles, que se muestre caminos outdoors, sea monocromático, calles con imágenes de satélite o mapa especializado para navegación y hasta un estilo en blanco que se utilizara para el mapa de interiores de este proyecto. Para cargar el mapa del interior se debe subir a Mapbox Studio el archivo .TIF generado por QGIS y en la Figura 8 se visualiza como el mapa de interior ya se muestra dentro estilo y por lo tanto ya está listo para ser usado en la aplicación Android o Web mediante el URL que es proporcionando desde Mapbox Studio y con el Token de acceso a para el mapa.

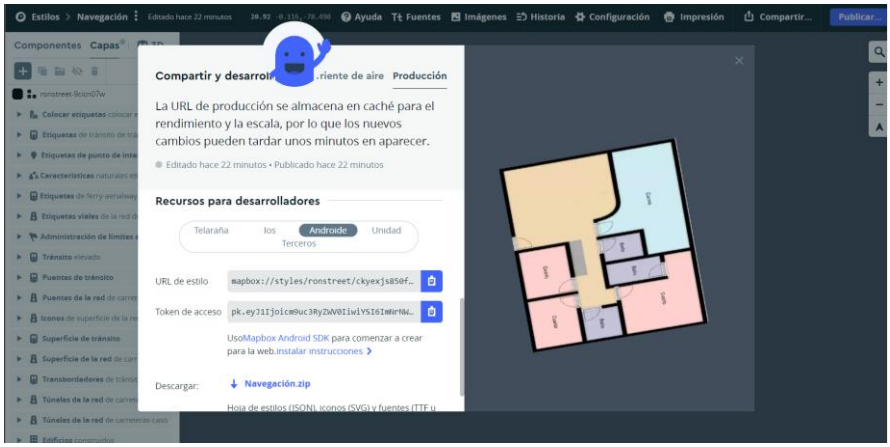


Figura 8. URL del estilo de mapa para utilizar en el aplicativo móvil y Web.

3.2.2 Base de Datos NoSQL Firebase

Para el almacenamiento de datos se ha tomado como información el nombre del punto, la latitud, longitud y las conexiones con otros puntos. Toda la información será almacenada dentro de una base de datos NoSQL, en la Tabla 6 se hace una comparación entre los 2 tipos de base de datos que maneja Firebase, en las cuales se tiene a RealTime Database y Cloud Firestore.

La primera tiene mucho más tiempo de uso en producción de aplicaciones, la información que se guarda es como un gran árbol JSON y también proporciona soporte sin conexión para clientes IOS y Android. En cambio, Cloud Firestore se trata de una base de datos no SQL que almacena los datos como colecciones de documentos e igualmente da soporte sin conexión agregando la Web. En términos generales Cloud Firestore es una mejora de su base de datos RealTime Database proporcionado mejor estructura de datos, para la inserción, eliminación, actualización y transacciones de datos. Los valores de costo y escalabilidad son ventajas que tienen Cloud Firestore, pero ambas opciones de base de datos pueden trabajar en conjunto. Pero el motivo principal para usar RealTime Database se trata del rendimiento rápido en lectura de los datos debido a su latencia extremadamente baja.

Tabla 6. Cuadro comparativo de RealTime Database y Cloud Firestore.

	RealTime Database	Cloud Firestore
Tipo de dato	Gran árbol JSON	Colección de documentos
Compatibilidad sin conexión	Android y IOS	Android, IOS y Web
Estado Conexión Cliente	Permite	No admitido nativamente
Rendimiento y rentabilidad	Mejor rentabilidad	Mejor rendimiento

(Firebase, 2022)

En la Tabla 7 se muestra los datos almacenados y requeridos para el funcionamiento tanto del aplicativo móvil y web.

Tabla 7. Información almacenada en la base de datos NoSQL.

Dato	Descripción
ID	Identificador único del marcador
Nombre	Nombre de marcador
Latitud	Punto de latitud para su geolocalización en Mapbox
Longitud	Punto de longitud para su geolocalización en Mapbox
Lugares	Conexión del marcador hacia otros marcadores.

3.2.3 Interfaz Gráfica del aplicativo móvil y web.

Se ha desarrollado una sola interfaz que se puede visualizar en la Figura 9, la cual estará integrado por 2 spinner para el despliegue de los puntos almacenados en la base de datos, de un mapView de Mapbox que permitirá mostrar el mapa personalizado del interior, un botón flotante para la lectura del código QR y un botón que desplegará la ruta corta de un punto a otro.

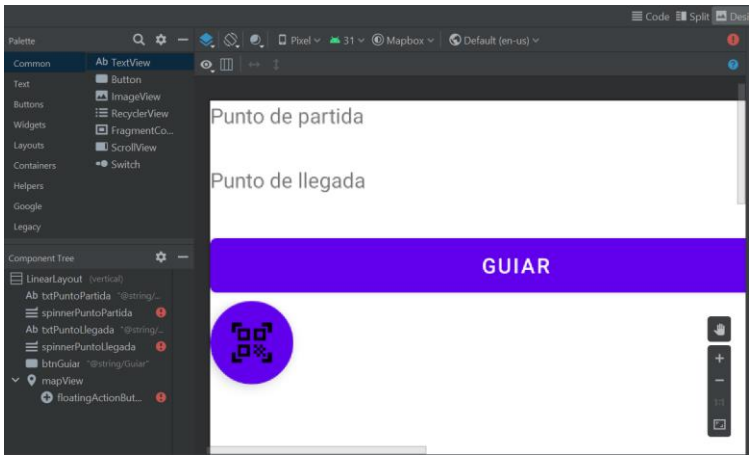


Figura 9. Desarrollo de la interfaz gráfica del aplicativo móvil.

Para el aplicativo móvil se utilizó Handlebars, que se trata de un motor de plantillas que permite hacer una página web sencilla en este caso para el CRUD de los lugares que almacenará como información en la base de datos. Como diseño se tendrá un encabezado que se mostrará en todas las páginas, el cuerpo donde se enlista todos los lugares que se ya encuentran almacenados en la base de datos con botones para editar y eliminar esos lugares de la base de datos. En la Figura 10 y la Figura 11 respectivamente se visualiza el código JavaScript y vista de la interfaz en la raíz de la página Web, el servidor para la ruta raíz envía la plantilla index.hbs pero también se manda un dato llamado Lugar, el dato enviado por el servidor es una lista y para poder visualizar esos datos dentro la index.hbs se hace uso del #each el cual va haciendo un recorrido de la lista y como los datos enviados se manejan como Json se puede obtener cada dato con su nombre respectivo por lo tanto se va llenará la tabla llamando al nombre del dato que en este caso es nombre, longitud, latitud y además se añade un botón con el id de cada Lugar.

```
<div class="card-body">
    {{#if Lugar}}
     <table class="table table-striped table-dark">
        <thead>
                <tr>
                <th scope="col">Nombre</th>
                <th scope="col">Longitud</th>
                <th scope="col">Latitud</th>
                <th scope="col">Opciones</th>
                </tr>
        </thead>
        <tbody>
            {{#each Lugar}}
                <tr>
                    <td>{{nombre}}</td>
                    <td>{{longitud}}</td>
                    <td>{{latitud}}</td>
                    <td><a href="/editar-marcador/{{@key}}" class="btn
                    btn-success"style="margin-right: 10px;">Editar</a>
                    <a href="/eliminar-marcador/{{@key}}" class="btn
                    btn-danger">Eliminar</a>
                    </td>
```

Figura 10. Código fuente de la interfaz principal de la página Web con Handlebars.

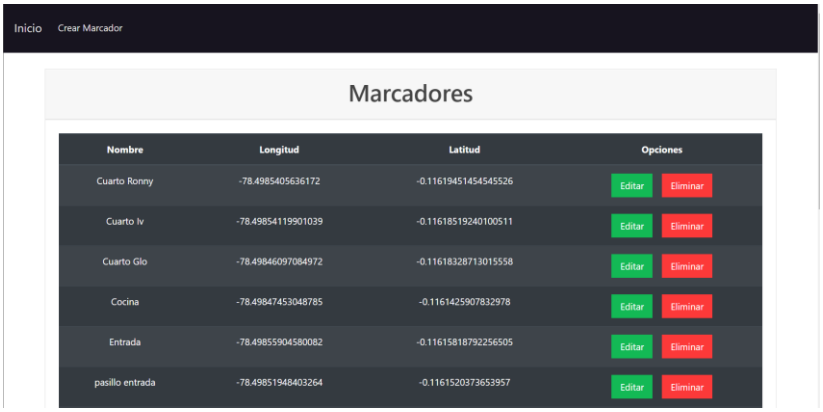


Figura 11. Interfaz gráfica del aplicativo web desarrollado con Handlebars.

El editar-marcador.hbs contiene el mismo diseño de la página principal, pero contiene un elemento para la visualización del mapa, y un formulario para edición del marcador previamente seleccionado desde la raíz principal y que el servidor estará enviando la información necesaria del marcador para visualizarlo en el formulario y en el mapa, así se muestra en la Figura 12 el código para añadir esos datos al formulario y el marcador se dibujará dentro del mapa.

```
<form action="/editar-marcador/{{id}}" method="post">

        <div class="form-group">
            <input type="text" name="nombre" value="{{nombre}}"
            placeholder="nombre" class="form-control" autofocus>
        </div>
        <div class="form-group">
            <input type="text" id='latitud' name="latitud"
            value="{{latitud}}" placeholder="latitud"
            class="form-control" readonly>
        </div>
        <div class="form-group">
            <input type="text" id='lontigud' name="longitud"
            value="{{longitud}}" placeholder="longitud"
            class="form-control"readonly>
        </div>
    <script>
```

Figura 12. Parte del código fuente editar-marcador.hbs.

En la Figura 13 se visualiza la página para la ruta editar-marcador en donde se muestra el marcador en el mapa, la información del marcador en el formulario para poder editarlo y un checkbox para seleccionar las conexiones con otros marcadores.

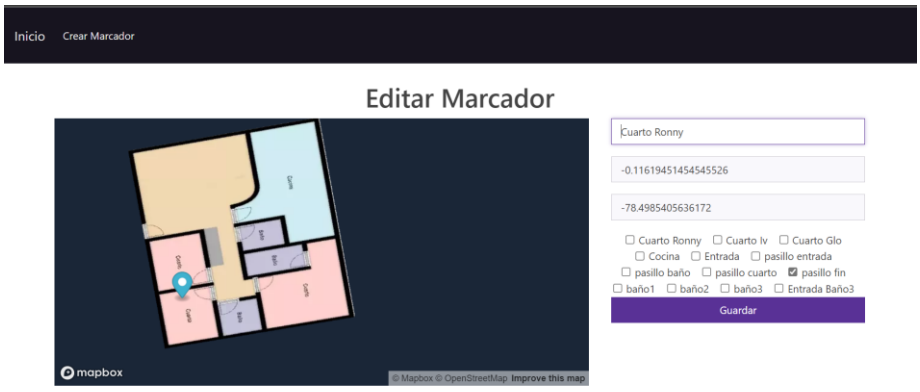


Figura 13. Interfaz gráfica del editar-marcador.hbs.

3.3 DESARROLLO

Para la fase de desarrollo se siguió el orden de iteraciones establecido en la fase de planeación para el cumplimiento de los requerimientos obtenidos a través de las historias de usuario.

3.3.1 Configuraciones previas antes de la realización de las iteraciones

Aplicativo Android

Para ello se ha utilizado la herramienta de Android Studio en la cual se ha implementado los siguientes repositorios que son Firebase, Mapbox, Google guava y Zxing como se puede ver en la Figura 14 está implementado en el archivo gradle de la aplicación.

```
dependencies {
    implementation 'com.mapbox.mapboxsdk:mapbox-android-sdk:9.6.2'
    implementation 'com.mapbox.mapboxsdk:mapbox-android-plugin-markerview-v9:0.4.0'
    implementation 'com.mapbox.mapboxsdk:mapbox-android-plugin-annotation-v9:0.9.0'

    implementation 'com.google.firebase:firebase-database'
    implementation platform('com.google.firebase:firebase-bom:28.4.1')
    implementation 'com.google.firebase:firebase-analytics'

    implementation 'com.mapbox.mapboxsdk:mapbox-sdk-turf:5.8.0'

    implementation 'com.journeyapps:zxing-android-embedded:4.2.0'
    implementation 'androidx.appcompat:appcompat:1.0.2'

    implementation 'androidx.appcompat:appcompat:1.3.1'
    implementation 'com.google.android.material:material:1.4.0'
    implementation 'androidx.constraintlayout:constraintlayout:2.1.0'
    implementation 'com.google.firebase:firebase-crashlytics-buildtools:2.8.1'

    implementation("com.google.guava:guava:31.0.1-android")
```

Figura 14. Dependencias utilizadas para el aplicativo móvil Android.

Los permisos requeridos para el aplicativo móvil se visualizan en la Figura 15, las cuales son el acceso al internet para poder hacer la carga de base de datos de Firebase, y para la utilización de Mapbox se requiere de los permisos para la ubicación aproximada y precisa esto deberá estar dentro del AndroidManifest.xml del proyecto de Android Studio.

```
<uses-sdk tools:overrideLibrary="com.google.zxing.client.android" />

<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
<uses-permission android:name="android.permission.INTERNET" />
```

Figura 15. Permisos requeridos para el aplicativo móvil.

Firebase

Para utilizar la base de datos de Firebase se necesita ingresar a la plataforma Web y luego ingresar a la consola de Firebase para el control de la base de datos, primeramente, se necesita crear el proyecto dentro la consola, y se visualiza en la Figura 16 cuando el proyecto ya está creado permitiendo ver nombre del proyecto denominado Tesis, y la parte inferior se encuentran las Aplicaciones de desarrollo conectadas al proyecto en este caso se muestra el TesisMapbox que representa la conexión al aplicativo móvil desarrollado. Una de las ventajas de Firebase es la analítica que se puede visualizar desde la

consola con lo cual se puede obtener información del uso de los datos por parte de los aplicativos asociados al proyecto.

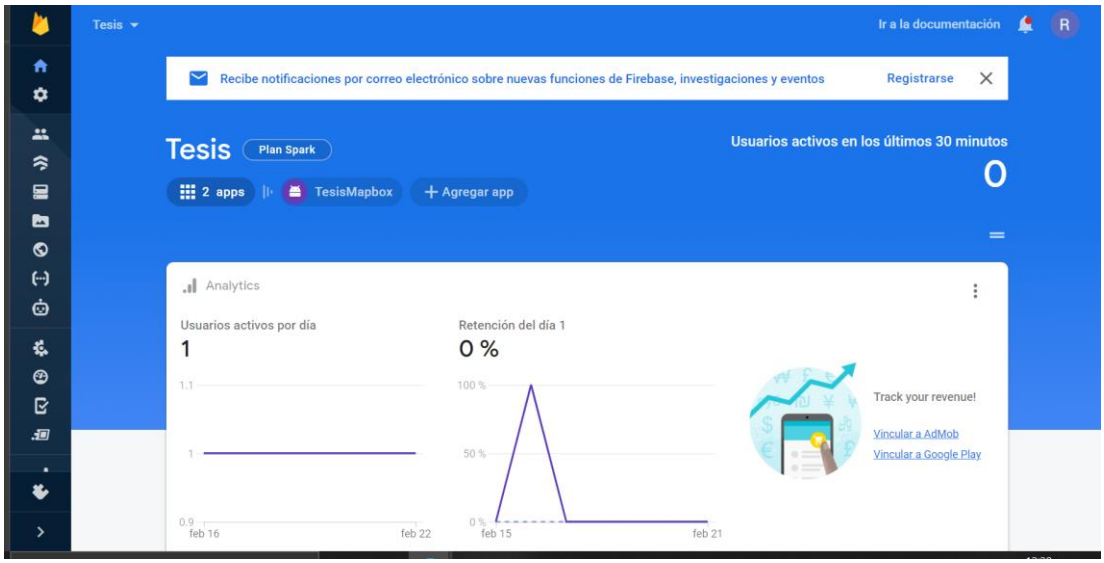


Figura 16. Ventana principal del proyecto para la utilización de la base de datos de Firebase.

Dentro de Firebase existe una guía de implementación del SDK de Firebase para Android en donde se pedirá el nombre del paquete del aplicativo y un sobrenombre para reconocerlo dentro de la consola con estos datos al registrar se generará una archivo tipo .Json con la configuración para ser usado dentro del aplicativo y se puede visualizar en la Figura 17 una parte de la información que este archivo contiene, en este caso se muestra la dirección URL de la base de datos, el nombre identificador del proyecto y la información cliente contiene los datos del aplicativo móvil y también se muestra el nombre del paquete solicitado anteriormente del aplicativo móvil.


```
{
  "project_info": {
    "project_number": "651824548662",
    "firebase_url": "https://tesis-459e4-default-rtdb.firebaseio.com",
    "project_id": "tesis-459e4",
    "storage_bucket": "tesis-459e4.appspot.com"
  },
  "client": [
    {
      "client_info": {
        "mobilesdk_app_id": "1:651824548662:android:0f20ac7ac8e58ca46a8197",
        "android_client_info": {
          "package_name": "com.example.firebase"
        }
      },
```


Figura 17. Archivo Json de configuración de Firebase para Android Studio.

En la Figura 18, se visualiza que en Android Studio el archivo. Json de configuración de la base de datos deberá ser añadida al proyecto al nivel de la App (A) y el código para inicializar la base de datos se muestra en la Figura 18 (B) en el que se implementa la inicialización de FirebaseApp y la obtención de la instancia predeterminada del FirebaseDatabase y la obtención del objeto DatabaseReference para el nodo raíz de la base de datos.

(A) (B)

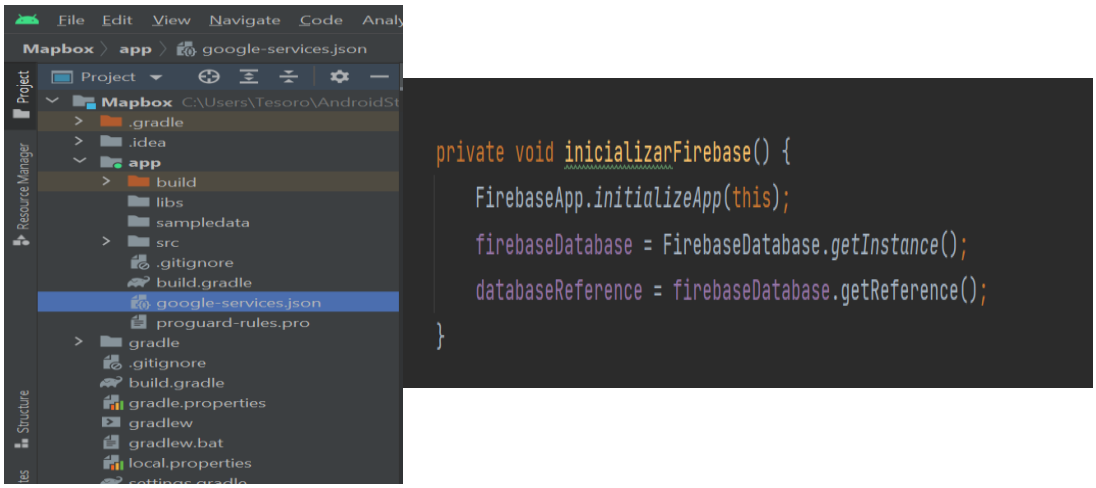


Figura 18. Directorio del archivo Json (A) y código fuente de inicialización de Firebase en Android Studio (B).

La base de datos de Firebase permitirá dentro del aplicativo de Android obtener los datos de los lugares que pueden presentarse dentro del mapa interiores, de tal manera se visualiza en la Figura 19 la función listarLugarLlegada el cual obtendrá los datos en un modelo clase Lugar, esta clase contiene el nombre, latitud y longitud de cada elemento y estos datos se ingresará dentro de un spinner los lugares que la persona podrá seleccionar como punto de llegada o partida.

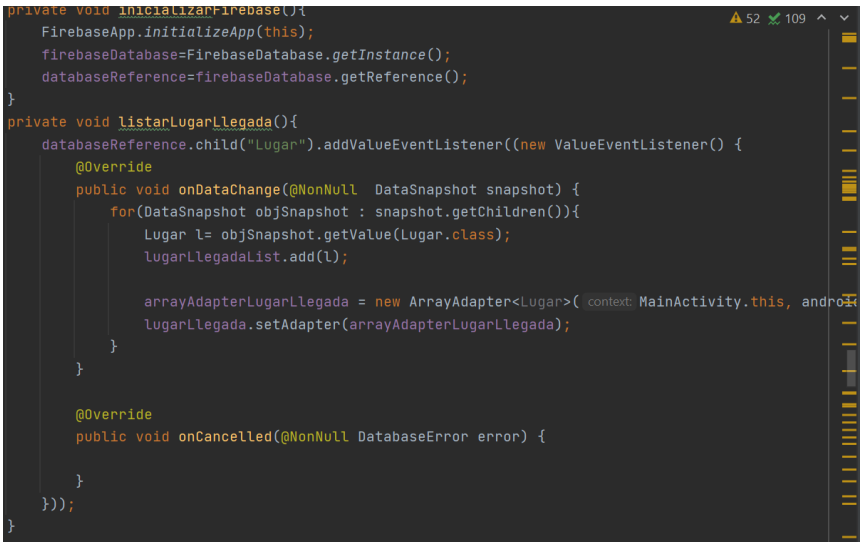


Figura 19. Código fuente de obtención y llenado de los datos de lugares de ida.

Aplicativo Web

Node JS

El servidor se ha implementado en Node JS, se escogió como parte del back-end por la escabilidad, la gestión de múltiples conexiones al mismo tiempo. Además de permitir una simple comunicación al no tener subprocesos, pero que permite aprovechar los múltiples núcleos del entorno y compartir sockets entre procesos. Node JS tiene un administrador de paquetes llamado nmp el cual permitirá integrar al servidor los distintos paquetes que se necesitan para trabajar en la página web como son: express, firebase-admin, express-handlebars. Así en la Figura 20 se muestran las dependencias instaladas.

Figura 20. Dependencias instaladas para el servidor Node JS.

Express permitirá tener las rutas, el control de peticiones desde las aplicaciones Web y en la Figura 21 se muestra la configuración de inicio del servidor en la que se tiene un puerto oyente que tendrá el servidor para las peticiones y además se especifica la utilización de express-handlebars como plantilla para usarse como ruta de vistas.

```
//settings
app.set('port', process.env.PORT || 4000);
app.set('views', path.join(__dirname,'views'));
app.engine('.hbs', engine({
    defaultLayout: 'main',
    extname: '.hbs',
    helpers:{
        igual:(arg1, arg2, options) =>{
            if(arg1==arg2){
                return options.fn(this);
            }
            return options.inverse(this);
        }}

}));
```

Figura 21. Código fuente de configuración del servidor para la página Web.

En routes/index.js que se visualiza en la figura 22, se establece las rutas que el servidor proporcionará y la actividad que realizará para cada ruta y petición solicitada por ejemplo el router.get(‘/’) es la ruta raíz con una petición get para lo cual el servidor realiza la acción de búsqueda en la base de datos y que al obtener todo los datos de la base de datos renderiza el index.hbs enviando además los datos Lugar de la base de datos para que en la plantilla se pueda trabajar con esos datos.

```
router.get('/', (req,res)=>{
    ref.once('value',(snapshot)=>{
        const data = snapshot.val();
        res.render('index',{Lugar:data});
    });
});

router.post('/nuevo-marcador', (req,res)=>{
    const newPostRef =ref.push();

    // Get the unique key generated by push()
    const postId = newPostRef.key;

    var lugares = req.body.lugares;
    if(lugares==null){
        lugares="";
    }
```

Figura 22. Código fuente para el manejo de rutas y peticiones al servidor.

Firebase

Para el manejo de la base datos en la página Web se utilizó el paquete de firebase-admin y para la visualización del mapa de mapbox se utilizó una llamada simple de JavaScript que se utilizará en la plantilla requerida como lo es el crear mapa y editar mapa. En Firebase-admin se debe crear una instancia a la base de datos mediante un URL y certificado de JSON proporcionado desde la Consola de Firebase. Para esto en la Figura 23 se muestra el código para la conexión a la base de datos NoSQL.

```
const admin = require('firebase-admin');

const serviceAccount = require("../../
tesis-459e4-firebase-adminsdk-rgg32-053fff63bc.json");
admin.initializeApp({
    credential: admin.credential.cert(serviceAccount), databaseURL:'https://
    tesis-459e4-default-rtdb.firebaseio.com/'
});


const db = admin.database();

const ref=db.ref('Lugar');
```

Figura 23. Inicialización y referenciado de Firebase en el servidor Node JS.

Mapbox

En Mapbox se debe proporcionar un TOKEN de acceso y estilo de mapa personalizado que se integró el mapa de interior y dependiendo del caso de la plantilla solicitada se agrega un oyente que en el caso de crear un marcador el oyente añadirá un marcador en el mapa, para el caso de actualizar el oyente moverá el marcador actual dentro del mapa. La integración de los oyentes tiene funciones como Click, drawend y etc, que se visualiza en la Figura 24.

```
<script>
        mapboxgl.accessToken = 'pk.
        eyJ1Ijoicm9uc3RyZWV0IiwiYSI6ImNrNWVlcHFmYzI2
        b2IzZG8za2J1emVpNTMifQ.
        UwczwBRhFZRxTGRIEyWEfQ';
        const map = new mapboxgl.Map({
        container: 'map', // container ID
        style: 'mapbox://styles/ronstreet/
        ckyak3wqv2igc14qawmk03gj5', // style URL
        center: [-78.498463, -0.1161695], //
        starting position [lng, lat]
        zoom: 20.7 // starting zoom
        });

        const marker = new mapboxgl.Marker({
            draggable: true
            })
            .setLngLat([document.getElementById
            ("lontigud").value, document.
            getElementById("latitud").value])
            .addTo(map);
```

Figura 24. Implementación del token de Mapbox para utilización en la página Web.

3.3.2 Iteraciones

Sistema de marcado en el piso

Se utilizo como sistema de marcado de piso los códigos QR ya que mediante ellos se puede almacenar información que puede ser obtenida mediante un dispositivo móvil, además que permite la facilidad de utilizarlo como marcado en cualquier superficie y que es fácil de reconocer para al usuario.

Mapa de interiores aplicativo móvil

Para mostrar el mapa de interiores se utilizó Mapbox, en este se debe proporcionar las credenciales como el usuario y el TOKEN de acceso de la cuenta de Mapbox las cuales deberán estar en el gradle del proyecto. En (mapbox, 2022) se proporciona toda la información y los pasos a seguir para la implementación de mapa de Mapbox en Android Studio, después de seguir los pasos para poder visualizar el mapa personalizado dentro del mapa Mapbox se tendrá que integrar el URL de estilo que se proporcionó anteriormente desde Mapbox Studio y como se muestra en la Figura 25 estaría configurado el mapa de interiores y esto deberá estar en el onMapReady que es parte del ciclo de vida del MapView de Mapbox.

```
@Override
public void onMapReady(@NonNull MapboxMap mapboxMap) {
    mapboxMap.setStyle(new Style.Builder().fromUri("mapbox://styles/ronstreet/ckyewxzwy004u15s7qoeb8ln0")
        @Override
        public void onStyleLoaded(@NonNull Style style) {
```

Figura 25. Implementación del mapa personalizado en el MapView del aplicativo móvil.

Dentro de la clase OnStyleLoad estará el código principal de funcionamiento del aplicativo móvil en donde consta del llamado al evento de botón Guiar y el evento para la lectura del código QR, dentro de esta clase que se muestra en la Figura 26 está la implementación de la animación del MapView para que al

iniciar el aplicativo móvil este se inicie en la posición del mapa de interiores y además se puede definir las propiedades para del mapa como set de Zoom mínimo y máximo.

.

```
//Punto al que se desplegara en el mapa como inicio
CameraPosition position = new CameraPosition.Builder()
        .target(new LatLng( latitude: -0.1161695,  longitude: -78.498495))
        .zoom(20.27)
        .build();
mapboxMap.animateCamera(CameraUpdateFactory.newCameraPosition(position),  durationMs: 2000);

//Limite de Zoom del mapa
mapboxMap.setMaxZoomPreference(22);
```

Figura 26. Código de animación para redirigir al MapView a la posición del mapa de interiores.

Lectura del marcador en el piso en el aplicativo móvil

Se utilizo Zxing que es una librería que permite el procesamiento de imágenes de código de barras 1D/2D. En este caso específico su integración permitirá el reconocimiento de códigos QR que proporcionará el ID de lugar y posterior a ello obtener la información requerida para el funcionamiento del aplicativo móvil. Así se puede visualizar en la Figura 27, la creación de la instancia integrada que ejecutara el lector del código QR, en el onActivityResult controlará el funcionamiento del lector cuando el usuario cancele la lectura del código QR y lanzará una activity cuando detecte un código QR.

```
findViewById(R.id.floatingActionButtonQR).setOnClickListener(new View.
OnClickListener() {
    @Override
    public void onClick(View v) {
        new IntentIntegrator(MainActivity.this).initiateScan();
    }
});

@Override
protected void onActivityResult(int requestCode, int resultCode, @Nullable
Intent data) {
    IntentResult result = IntentIntegrator.parseActivityResult(requestCode,
    resultCode,data);
    if(result!=null){
        if(result.getContents()==null){
            Toast.makeText(this,"Cancelado",Toast.LENGTH_SHORT).show();
        }else {
            Toast.makeText(this,"El valor escaneado es:"+result.getContents().
            toString(),Toast.LENGTH_LONG).show();
            listarLugarIda1(result.getContents().toString());
        }
    }else{
        super.onActivityResult(requestCode, resultCode, data);
        super.onActivityResult(requestCode, resultCode, data);
    }
}
```

Figura 27. Código fuente para Llamada del activity para la lectura del código QR.

En la Figura 28 se muestra la interfaz del activity para la lectura de código QR en donde si el usuario retrocede sin la captura de un código QR regresará al activity principal con una variable resultado null y por lo tanto se informará con un mensaje de que la captura de código QR ha sido cancelado. Para el caso de que se obtenga la captura de un código QR se enviará el dato capturado

para seleccionarlo en el primer spinner como punto de partida para la ruta corta.

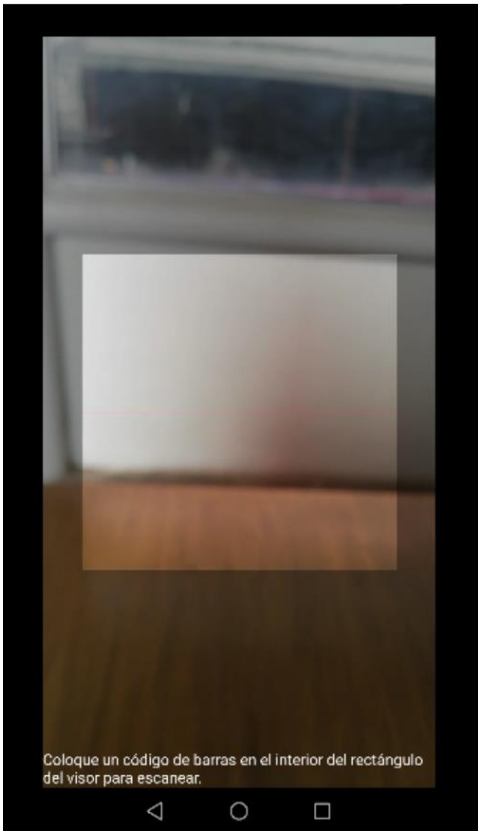


Figura 28. Visualización de la cámara para la búsqueda del código QR.

Visualización del marcador en el aplicativo móvil

Para marcar los puntos de la base de datos dentro del mapa se tiene una función llamada agregarMarcador que se puede visualizar en la Figura 29 la cual recibe como parámetros una clase Lugar y Mapboxmap, en donde desde la clase Lugar se obtiene los datos de longitud y latitud, que posteriormente se agrega el marcador dentro del mapa recibido como parámetro. Por lo tanto, esta función sirve tanto para agregar el marcador del lugar de ida y el de lugar de llegada.

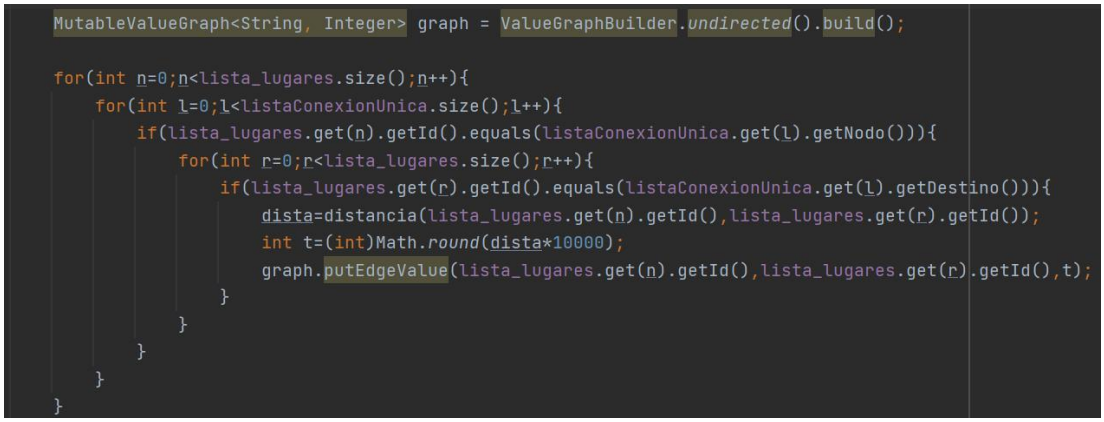
```
private void agregarMarcador(@NonNull Lugar lugar, @NonNull MapboxMap mapboxMap) {
    mapboxMap.addMarker(new MarkerOptions()
            .position(new LatLng(Double.parseDouble(lugar.getLatitud()), Double.parseDouble(lugar.getLongitud())))
            .title("Marcador"));

}
```

Figura 29. Función para la inserción de un marcador en el MapView de Mapbox.

Algoritmo de ruta corta

Para la ruta corta se implementó el algoritmo de Dijkstra, el cual es uno de los algoritmos más populares y que se trata de una búsqueda codiciosa en donde el grafo que se debe generar de los Lugares junto a sus conexiones deberá

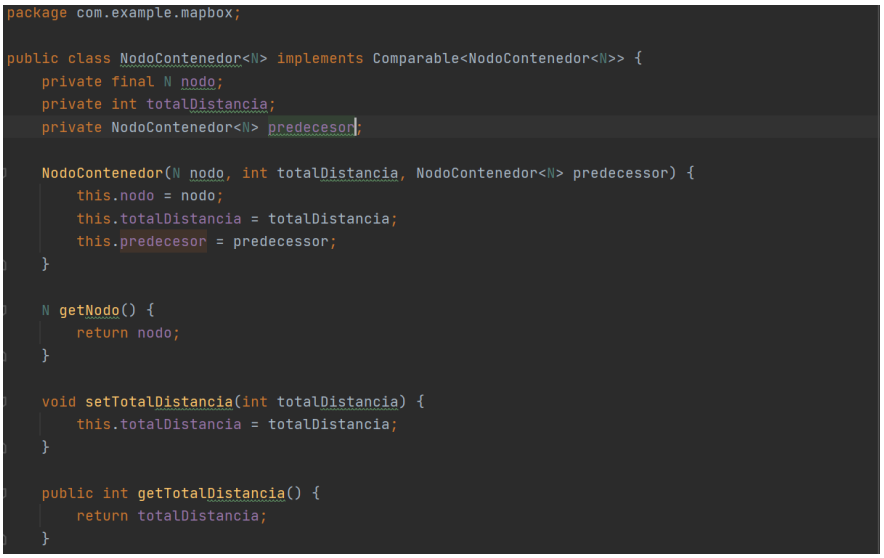
ser un gráfico ponderado y que sus pesos no podrán ser negativos. Para utilizar este algoritmo se debe tener una estructura del grafo y se lo realizó mediante la librería de Google guava esta librería permitirá crear un grafo en donde los nodos serán los Lugares de la base de datos y sus pesos hacia a otros nodos será la distancia medida de un punto geográfico a otro. En la Figura 30 se visualiza que primero se debe crear el grafo mediante la librería de guava, en donde se hace una instancia del grafo en el cual tendrá los siguientes tipos de datos el String que será el nombre e identificador del nodo, y un dato tipo Integer que almacenará el peso. Este código necesita los parámetros que son el tipo de estructura de dato para el grafo y si el grafo es dirigido o no dirigido, pero para este caso no lo es por lo tanto el orden de creación de conexiones del grafo no es importante el orden de conexión de los nodos.



```
MutableValueGraph<String, Integer> graph = ValueGraphBuilder.undirected().build();

for(int n=0;n<lista_lugares.size();n++){
    for(int l=0;l<listaConexionUnica.size();l++){
        if(lista_lugares.get(n).getId().equals(listaConexionUnica.get(l).getNodo())){
            for(int r=0;r<lista_lugares.size();r++){
                if(lista_lugares.get(r).getId().equals(listaConexionUnica.get(l).getDestino())){
                    dista=distancia(lista_lugares.get(n).getId(),lista_lugares.get(r).getId());
                    int t=(int)Math.round(dista*10000);
                    graph.putEdgeValue(lista_lugares.get(n).getId(),lista_lugares.get(r).getId(),t);
                }
            }
        }
    }
}
```

Figura 30. Código fuente para la creación del grafo de los lugares para la ruta corta.

Para obtener las distancias y los predecesores de un Nodo para la ruta corta se tiene una clase que denominada NodoContenedor y en la Figura 31 se muestra que dentro de esta clase tiene las siguientes variables denominadas Nodo, totalDistancia y un NodoContenedor con sus respectivos getters y setters y esta clase es la que guardará la información al realizar la función de la ruta corta.

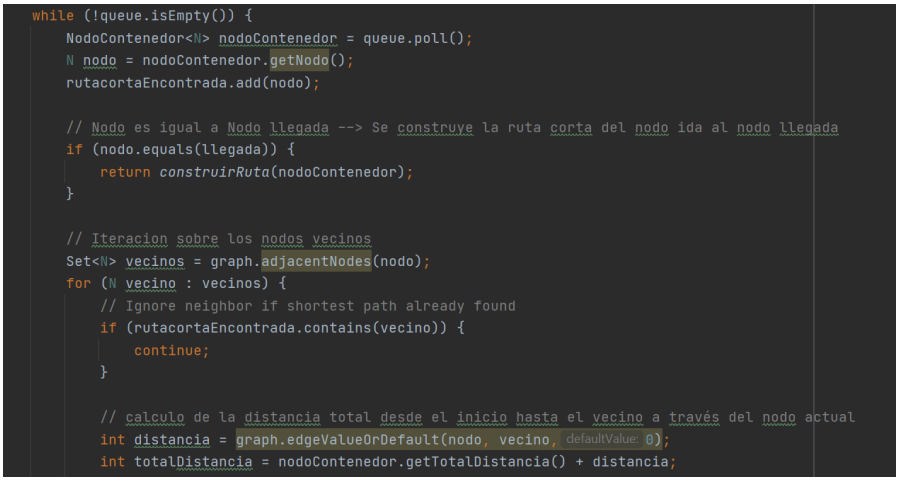


```java
package com.example.mapbox;

public class NodoContenedor<N> implements Comparable<NodoContenedor<N>> {
    private final N nodo;
    private int totalDistancia;
    private NodoContenedor<N> predecesor;

    NodoContenedor(N nodo, int totalDistancia, NodoContenedor<N> predecessor) {
        this.nodo = nodo;
        this.totalDistancia = totalDistancia;
        this.predecesor = predecessor;
    }

    N getNodo() {
        return nodo;
    }

    void setTotalDistancia(int totalDistancia) {
        this.totalDistancia = totalDistancia;
    }

    public int getTotalDistancia() {
        return totalDistancia;
    }
```

Figura 31. Código fuente de la Clase NodoContenedor para la información de distancia y ruta corta de un nodo.

Para la solución del algoritmo de ruta corta implementado con Dijkstra y PriorityQueue se tomó como referencia el trabajo de (Woltmann, 2020), en el que describe el funcionamiento del algoritmo de Dijkstra con algunos métodos de uso como PriorityQueue, con la clase Treeset y hasta con un método Fibonacci.

El algoritmo implementado para ruta corta ha sido mediante el PriorityQueue que hará de comparador y que mantiene al elemento más pequeño a la cabeza (Oracle, 2022),. Luego se hará un recorrido para cada nodo para la búsqueda de la ruta corta que se guardará dentro del NodoContenedor, en la Figura 32 se denota un bucle While que seguirá realizando el bucle mientras la cola no esté vacía. Se tiene también un condicional mientras el nodo Contenedor no sea igual al nodo de llegada seguirá en el bucle.



```java
while (!queue.isEmpty()) {
    NodoContenedor<N> nodoContenedor = queue.poll();
    N nodo = nodoContenedor.getNodo();
    rutacortaEncontrada.add(nodo);

    // Nodo es igual a Nodo llegada --> Se construye la ruta corta del nodo ida al nodo llegada
    if (nodo.equals(llegada)) {
        return construirRuta(nodoContenedor);
    }

    // Iteracion sobre los nodos vecinos
    Set<N> vecinos = graph.adjacentNodes(nodo);
    for (N vecino : vecinos) {
        // Ignore neighbor if shortest path already found
        if (rutacortaEncontrada.contains(vecino)) {
            continue;
        }

        // calculo de la distancia total desde el inicio hasta el vecino a través del nodo actual
        int distancia = graph.edgeValueOrDefault(nodo, vecino, defaultValue: 0);
        int totalDistancia = nodoContenedor.getTotalDistancia() + distancia;
```

Figura 32. Código fuente para la búsqueda de la ruta corta de un punto de inicio y punto de llegada.

En la Figura 33 se sigue con la iteración de los vecinos de nodo evaluado en donde si la ruta encontrada ya contiene el nodo vecino se ignora la iteración

de ese nodo vecino, si el nodo vecino no se encuentra en la ruta encontrada se compara la distancia del nodo vecino a nodo evaluado y si el vecino contenedor no se encuentra dentro de la ruta se agregará el vecino, la distancia nueva y el nodo contenedor, en cambio, si el nodo fue ya descubierto se compara la distancia total del nodo actual al vecino contenedor y si este es menor se enviará la nueva distancia más corta y el nuevo contenedor.

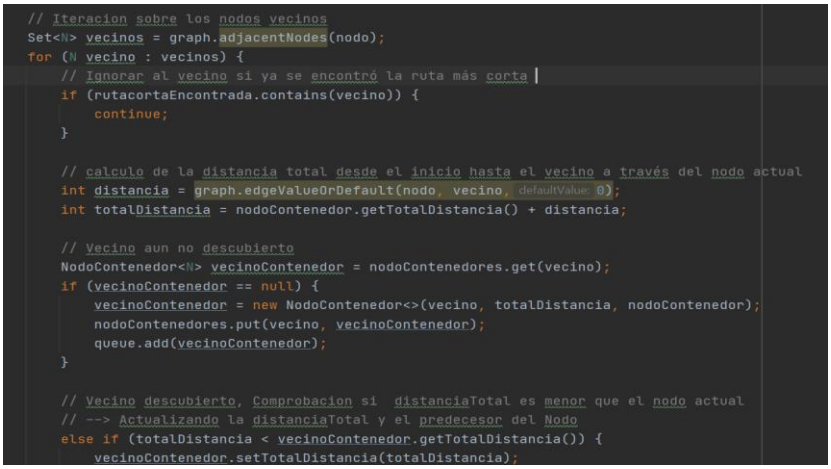


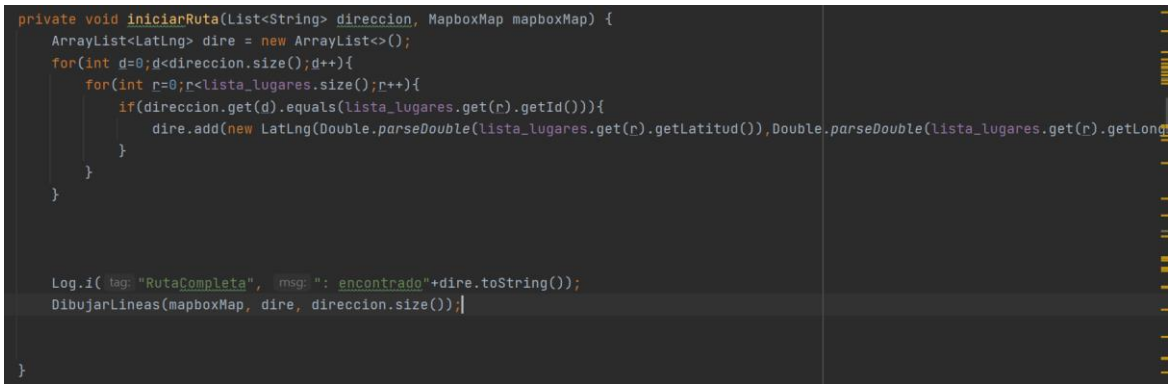
Figura 33. Código fuente donde se compara las distancias y nodo predecesor.

Para obtener la ruta corta del nodo inicial al de llegada se hace el llamado a la función construirRuta, en la Figura 34 se muestra el código que realiza el recorrido del nodoContenedor mientras sea diferente a null y en cada recorrido se añade al path el nodo.contenedor y nodoContenedor cambia a nodo predecesor.

```
private static <N> List<N> construirRuta(NodoContenedor<N> nodoContenedor) {
    List<N> path = new ArrayList<>();
    while (nodoContenedor != null) {
        path.add(nodoContenedor.getNodo());
        nodoContenedor = nodoContenedor.getPredecesor();
    }
    Collections.reverse(path);
    return path;
}
```

Figura 34. Código fuente para añadir la ruta corta de un nodo.

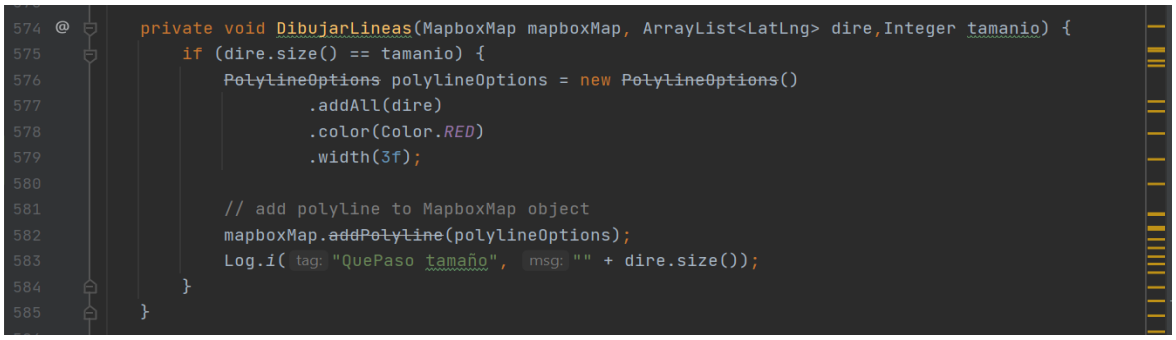
Para mostrar el camino generado como ruta corta se utilizó una función llamada IniciarRuta que permitirá buscar los datos de latitud y longitud de un punto desde la base de datos y mediantes los puntos que algoritmo de Dijkstra determine como puntos de conexión desde un punto a otro se almacenará en un arreglo String. Por lo tanto, en la Figura 35 se muestra el código que permite recorrer el arreglo de String con los puntos para almacenar en un nuevo arreglo con la latitud y longitud de cada punto y la información necesaria para su visualización dentro del mapa es enviado al dibujarLinea().



```
private void iniciarRuta(List<String> direccion, MapboxMap mapboxMap) {
    ArrayList<LatLng> dire = new ArrayList<>();
    for(int d=0;d<direccion.size();d++){
        for(int r=0;r<lista_lugares.size();r++){
            if(direccion.get(d).equals(lista_lugares.get(r).getId())){
                dire.add(new LatLng(Double.parseDouble(lista_lugares.get(r).getLatitud()),Double.parseDouble(lista_lugares.get(r).getLong
            }
        }
    }



    Log.i( tag: "RutaCompleta",  msg: ": encontrado"+dire.toString());
    DibujarLineas(mapboxMap, dire, direccion.size());


}
```

Figura 35. Búsqueda de los puntos y trazado de los puntos en el mapa interior.

Por último se tiene la función dibujarLineas e iniciarRuta, en la primera función que se muestra en la Figura 36 permitirá dibujar la ruta del punto de ida al punto de destino, para ello recibirá como parámetros el mapa donde se dibujara la líneas y una lista de arreglo de LatLng generado de la ruta corta que se dibujara dentro del mapa mediante addPolyline que requiere una variable tipo PolylineOptions tendrá parámetros de configuración como el color de las líneas, el tamaño y el arreglo de puntos el cual permitirá dibujar una línea entre todos los puntos de la lista de arreglo.



```
574 @   private void DibujarLineas(MapboxMap mapboxMap, ArrayList<LatLng> dire,Integer tamanio) {
575         if (dire.size() == tamanio) {
576             PolylineOptions polylineOptions = new PolylineOptions()
577                     .addAll(dire)
578                     .color(Color.RED)
579                     .width(3f);
580
581             // add polyline to MapboxMap object
582             mapboxMap.addPolyline(polylineOptions);
583             Log.i( tag: "QuePaso tamaño",  msg: "" + dire.size());
584         }
585     }
```

Figura 36. Opciones para el trazado de líneas entre puntos del mapa.

Ingreso de nuevos marcadores aplicativo web

Para el ingreso de nuevos marcadores desde el aplicativo web se toma los datos requeridos para el almacenamiento en la base de datos, en la Figura $$ se visualiza el código para mostrar el punto del marcado dentro del mapa de interiores.

En la Figura 37 se visualiza el código para la creación de un nuevo marcador desde el aplicativo web en la base de datos de Firebase.

```
router.post('/nuevo-marcador', (req,res)=>{
    const newPostRef =ref.push();

    // Get the unique key generated by push()
    const postId = newPostRef.key;

    var lugares = req.body.lugares;
    if(lugares==null){
        lugares="";
    }
    console.log(lugares);
    const nuevoMarcador={
        id: postId,
        nombre: req.body.nombre,
        latitud: req.body.latitud,
        longitud: req.body.longitud,
        lugares: lugares
    };

    newPostRef.set(nuevoMarcador);
    //newPostRef.push(nuevoMarcador);
    console.log(postId);
    console.log(req.body);
    res.redirect('/');
});
```

Figura 37. Código para la creación de un nuevo marcador.

Eliminación de marcadores aplicativo web

Para la eliminación del marcador se obtiene el ID único de marcador y posterior a ello se manda a la ruta eliminar-marcador/:id la cual es la recibe como parámetro el id único del marcador y redirigiendo a la página principal en la Figura 38 se muestra el código para la ruta eliminar.

```
router.get('/eliminar-marcador/:id',(req,res)=>{
    db.ref('Lugar/'+req.params.id).remove();
    console.log(req.params.id);
    res.redirect('/');
});
```

Figura 38. Eliminación de un marcador.

Actualización de marcadores aplicativo web

En la Figura 39 se visualiza el redireccionamiento a la ruta get: editar-marcador que para mostrar la información del marcador requiere realizar una búsqueda en la base de datos mediante la id enviada como parámetro.

Figura 39. Obtención de los datos del marcador para actualizar.

Para la actualización ya del marcador se envía como parámetros el id, el nombre, latitud, longitud, los marcadores con los que se conecta a la ruta post: editar-marcador/:id en donde podemos visualizar en la Figura 40 el código de actualización de marcador en la base de datos.

Figura 40. Código para la actualización de información de un marcador.

QRCODE

Se trata de una dependencia que se implementará dentro del servidor y que permitirá crear la imagen QR del dato seleccionado como identificador para el aplicativo móvil. Para ello se debe hacer la instalación del paquete de la siguiente manera npm install -save qrcode, esto permitirá tener una librería para crear una función que generará una imagen QR del marcador seleccionado. En la página Web el ID del lugar es el dato que se almacena dentro de una imagen QR y que se podrá descargar desde la página Web.

En el archivo routes/index.js se ingresó una nueva ruta para la creación del QR y se muestra en la Figura 41 como utilizar la dependencia qrcode para la creación del código QR y que posteriormente se redirigirá a otra ruta donde se visualizará la imagen QR para que pueda ser descargada por parte del cliente y poder ubicarlo en la posición dentro la infraestructura. En ella se tiene que llamar al toDataURL( );en donde se enviará como parámetro el ID del lugar, otro elemento es la versión del código que va desde el 1 hasta el 40, la diferencia de las versiones se trata del número de puntos blancos, negros que contienen y del tamaño de la imagen QR.

```
const generarQR= require('qrcode');

router.get('/generarQR/:id',(req,res)=>{
    const qrID=req.params.id;
    generarQR.toDataURL(qrID,{ version: 5 },function(err,url){
        if(err) res.send('Código QR no disponible')
        console.log(url);
        res.render('codigoQR',{
            QR:url,});
    })
});
```

Figura 41. Código fuente para creación del código QR.

En la Figura 42, se visualiza la interfaz gráfica y la forma en cómo se representará la imagen QR que sirva como marcado en el piso.

Figura 42. Interfaz gráfica de la visualización del código QR.

3.4 PRUEBAS

Para la fase de pruebas se realizaron pruebas funcionales tanto para el aplicativo móvil y web. En la Tabla 8, Tabla 9, Tabla 10 y Tabla 11 se muestran las pruebas realizadas y con los resultados obtenidos.

Tabla 8. Pruebas Funcionales 1-2.

Pruebas Funcionales		
Número	**1**	**2**
Caso de prueba	Visualización del mapa de interiores en el aplicativo móvil	Conexión a la base de datos en el aplicativo móvil
Descripción	El objetivo de la prueba es verificar que el mapa se muestre correctamente dentro del aplicativo móvil	El objetivo de la prueba es verificar que el aplicativo móvil se conecta a la base de datos de Firebase
Precondiciones	Integración del mapa de interiores en Mapbox Studio	Configuración y permisos para el aplicativo
Resultado esperado	El aplicativo debe mostrar el mapa de interiores y su tamaño debe ser visible para cualquier persona.	El aplicativo móvil al iniciar debe conectarse a la base de datos y desplegar la información.
Resultado obtenido	Se mostro correctamente el mapa de interiores	El aplicativo móvil se conectó correctamente con la base de datos
Estado	Revisado	Revisado

Tabla 9. Pruebas Funcionales 3-4.

Pruebas Funcionales		
Número	**3**	**4**
Caso de prueba	Detección del código QR	Mostrar la ruta corta del de punto de partida al punto de llegada
Descripción	El objetivo de la prueba es verificar que el aplicativo móvil detecte el código QR correctamente	El objetivo de la prueba es verificar que la ruta corta visualizada sea la correcta
Precondiciones	El dispositivo móvil debe tener cámara y permisos	Haber detectado el código QR y seleccionado el punto de llegada
Resultado esperado	El aplicativo móvil detecte el código QR y seleccione automáticamente el punto de partida	El aplicativo móvil debe mostrar la ruta más corta entre los 2 puntos

Resultado obtenido	Se detectó el código QR de manera correcta y se selecciona de manera automática el punto de partida	Se visualiza la ruta corta correctamente
Estado	Revisado	Revisado

Tabla 10. Pruebas Funcionales 5-6.

Pruebas Funcionales		
Número	**5**	**6**
Caso de prueba	Creación de nuevos marcadores	Eliminación de marcadores
Descripción	El objetivo de la prueba es verificar que se almacenen correctamente en la base de datos los marcadores	El objetivo de la prueba es verificar que se eliminen correctamente en la base de datos los marcadores
Precondiciones	Ninguna	Ninguna
Resultado esperado	El aplicativo web almacene correctamente el marcador con la información proporcionada	El aplicativo web elimine el marcador y actualice o redirige hacia la página principal
Resultado obtenido	El aplicativo web almacenó correctamente la información y creo el id Único para el marcador	El aplicativo web elimino correctamente el marcador y actualizo la página principal
Estado	Revisado	Revisado

Tabla 11. Pruebas Funcionales 7-8.

Pruebas Funcionales		
Número	**7**	**8**
Caso de prueba	Actualización de marcadores	Generar código QR
Descripción	El objetivo de la prueba es verificar que se actualicen correctamente en la base de datos los marcadores	El objetivo de la prueba es verificar que se genere correctamente el código QR
Precondiciones	Ninguna	Tener información en la base de datos
Resultado esperado	La base de datos se actualiza correctamente con la nueva información	Se genera correctamente el código QR para su descargar o visualización del marcador seleccionado
Resultado obtenido	Se actualizo correctamente la información del marcador en la base de datos	Se generó y se visualiza correctamente el código QR del marcador seleccionado
Estado	Revisado	Revisado

4. CONCLUSIONES Y RECOMENDACIONES

4. CONCLUSIONES Y RECOMENDACIONES

4.1 CONCLUSIONES

Con el presente trabajo se logró implementar un sistema de navegación de interiores mediante una señalética marcada en el piso que será detectada por la aplicación móvil y esto permitirá al usuario reconocer su ubicación y guiarse desde ese punto hacia a su destino. Para lograrlo se necesitó una base de datos que permitiera almacenar la información de ubicación de cada imagen dentro de la infraestructura y que a través de una aplicación móvil reconociera estas imágenes que permitirán obtener la información almacenada en la base de datos y mostrarla en un mapa de interior como punto de ubicación del usuario y que mediante un algoritmo de ruta corta mostrará la ruta para llegar hacia su destino.

El código QR permitió tener un sistema integrado de señalética de marcado en el piso, debido a su facilidad de implementación y ubicación dentro una infraestructura y además facilitando a la aplicación móvil la detención de estas imágenes y que el usuario pueda reconocer fácilmente estos marcados dentro la infraestructura.

Se logró desarrollar una aplicación móvil que a través de la captura de una imagen QR, permitiera mostrar la ubicación dentro de un mapa interior utilizando un mapa personalizado en Mapbox y además con el uso del algoritmo de Dijkstra mostrar la ruta corta desde el punto escaneado hacia su destino.

Se implementó una aplicación Web mediante el lenguaje de JavaScript tanto para el Back-End y Front-End, que permitió realizar el ingreso, modificación y eliminación de marcadores que servirán como configuración para el aplicativo móvil y de esta manera obtener una aplicación Android que funcione correctamente.

.

4.2 RECOMENDACIONES

Para reconocer la posición del usuario dentro del mapa de interiores se utilizó como marcador de piso el código QR, esto implica tener que ubicar estos códigos QR en lugares visibles y en el que se pueda mantener su integridad sin que se dañe con el tiempo, sean alterados y no permitan al aplicativo obtener la información del código QR.

El aplicativo móvil realizado solo trabaja en dispositivos Android. Sin embargo, las herramientas utilizadas podrían ser utilizadas para establecer soporte con dispositivos IOS.

El aplicativo móvil podría implementar un sistema de voz que permita guiar al usuario dentro del establecimiento y a su vez se podría añadir otro sistema de posicionamiento de interiores para mejorar el rendimiento y efectividad del aplicativo para localización del usuario.

El aplicativo móvil y web podría mejorar el apartado de seguridad implementando restricciones en el acceso, en la protección de la información debido a que se corre el riesgo de ser utilizada de manera incorrecta.

BIBLIOGRAFÍA

Bibliografía

Aguilar Herrera, J. C., Plöger, P. G., Hinkenjann, A., Maiero, J., Flores, M., & Ramos, A. (2014). Pedestrian indoor positioning using smartphone multi-sensing, radio beacons, user positions probability map and IndoorOSM floor plan representation. *2014 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, 636–645. https://doi.org/10.1109/IPIN.2014.7275538

Baesens, B., Backiel, A., & vanden Broucke, S. (2015). Beginning Java Programming: The Object-Oriented Approach. Indianapolis: Wiley.

Basu, S. K. (2005). *DESIGN METHODS AND ANALYSIS OF ALGORITHMS*. PHI Learning.

Batistić, L., & Tomić, M. (2018). Overview of Indoor Positioning System Technologies. *MIPRO*.

Candra, A., Budiman, M. A., & Hartanto, K. (2020). Dijkstra's and A-Star in Finding the Shortest Path: a Tutorial. *2020 International Conference on Data Science, Artificial Intelligence, and Business Analytics (DATABIA)*, 28–32. https://doi.org/10.1109/DATABIA50434.2020.9190342

Capacho, J., & Nieto, W. (2017). *DISEÑO DE BASE DE DATOS.* Barranquilla: Universidad del Norte.

Climent, P. V. (4 de Noviembre de 2019). *¿Qué productos y servicios ofrece Mapbox?* Obtenido de MappingGIS SLU: https://mappinggis.com/2019/11/que-productos-y-servicios-ofrece-mapbox/

Cankaya, I. A., Koyun, A., Yigit, T., & Yuksel, A. S. (2015). Mobile indoor navigation system in iOS platform using augmented reality. *2015 9th International Conference on Application of Information and Communication Technologies (AICT)*, 281–284. https://doi.org/10.1109/ICAICT.2015.7338563

Chaudhery, M. H. (2021). *Smartphone-Based Detection Devices: Emerging Trends in Analytical Techniques*. Elsevier.

Chitra, L. P., & Satapathy, R. (2017). *Performance Comparison and Evaluation Of Node.Js And Traditional Web Server (IIS)*.

Descamps-Vila, L., Navarro, A. P., & Caralt, J. C. (2013). *Integración de un sistema de posicionamiento indoor en aplicaciones SIG para dispositivo móvil*.

Díaz, E., Pérez, M. C., Gualda, D., Villadangos, J. M., Ureña, J., & García, J. J. (2017). Ultrasonic indoor positioning for smart environments: A mobile application. *2017 4th Experiment@International Conference (Exp.at'17)*, 280–285. https://doi.org/10.1109/EXPAT.2017.7984382

Firebase. (2022). *Firebase Realtime Database*. Obtenido de firebase.google: https://firebase.google.com/docs/database

Flanagan, D. (2011). *JavaScript: The Definitive Guide.* O'Reilly Media.

García, A. F., Gómez, C., Sánchez, T., Redondo, A. D., Betancur, L., & Hincapié, R. C. (2015). Algoritmos de Radiolocalización basados en ToA, TDoA y AoA Radiolocation Algorithms based on ToA, TDoA and AoA. In *Revista Ingeniería y Región* (Vol. 14, Issue 2).

Hébuterne, S. (2016). *Android: guía de desarrollo de aplicaciones Java para smartphones y tabletas.* Barcelona: ENI.

Joyanes Aguilar, L. (2016). *Big Data, Análisis de grandes volúmenes de datos en organizaciones.* Mexico D.F: Alfaomega Grupo Editor.

Kastanakis, B. (2016). *Mapbox Cookbook*. Packt Publishing.

Letelier, P., & Penadés, C. (2012). Métodologías ágiles para el desarrollo de software: eXtreme Programming (XP). *Métodologías Ágiles Para El Desarrollo de Software: EXtreme Programming (XP)*.

Li, Z., Deng, Z., Liu, W., & Xu, L. (2013). A Novel Three-Dimensional Indoor Localization Algorithm Based on Multi-Sensors. In *Lecture Notes in Electrical Engineering* (Vol. 245, pp. 623–631). https://doi.org/10.1007/978-3-642-37407-4_58

Mamaeva, D., Afanasev, M., Bakshaev, V., & Kliachin, M. (2019). A multi-functional method of QR code used during the process of indoor navigation. *2019 International Conference on Engineering and Telecommunication (EnT)*, 1–4. https://doi.org/10.1109/EnT47717.2019.9030587

Manricks, G. (2013). *Instant Handlebars.js.* Birmingham: Packt Publishing Ltd .

mapbox. (2022). *Installation*. Obtenido de mapbox: https://docs.mapbox.com/android/maps/guides/install/

mapbox. (2022). *Our Vision*. Obtenido de Mapbox: https://www.mapbox.com/about/company/

Oracle. (2022). *Class PriorityQueue<E>*. Obtenido de docs.oracle: https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/util/PriorityQueue.html

Pozo Ruz, A., Ribeiro, A., García-Alegre, M. C., García, L., Guinea, D., & Sandoval, F. (2000). Sistema De Posicionamiento Global (Gps): Descripción, Análisis De Errores, Aplicaciones Y Futuro. *ETS Ingenieros de Telecomunicaciones. Universidad de Malaga*, 174. http://www.oocities.org/es/forogps/infografia/gps5.pdf

QGIS project. (2022). *2. Prólogo — documentación de QGIS Documentation* - . Obtenido de qgis.org: https://docs.qgis.org/3.16/es/docs/user_manual/preamble/foreword.html

Rantanen, J., Mäkelä, M., Ruotsalainen, L., & Kirkko-Jaakkola, M. (2018). Motion Context Adaptive Fusion of Inertial and Visual Pedestrian Navigation. *2018 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, 206–212. https://doi.org/10.1109/IPIN.2018.8533872

Rashid, M. T., Chowdhury, M. S. S., & Nawal, M. F. (2016). Composite indoor localization and positioning system based on Wi-Fi repeater technology. *2016 IEEE Region 10 Conference (TENCON)*, 2130–2133. https://doi.org/10.1109/TENCON.2016.7848402

Sarasa, A. (2019). *Introducción a las bases de datos NSQL clave-valor usando Redis.* Barcelona: UOC.

Shen, G., & Huang, X. (2011). *Advanced Research on Electronic Commerce, Web Application, and Communication*. Springer Science & Business Media.

Subedi, S., Gang, H.-S., Ko, N. Y., Hwang, S.-S., & Pyun, J.-Y. (2019). Improving Indoor Fingerprinting Positioning With Affinity Propagation Clustering and Weighted Centroid Fingerprint. *IEEE Access*, *7*, 31738–31750. https://doi.org/10.1109/ACCESS.2019.2902564

Wang, G., & Pan, B. (2012). *Passive RF Component Technology*.

Woltmann, S. (25 de Noviembre de 2020). *Dijkstra's Algorithm (with Java Examples)*. Obtenido de happycoders: https://www.happycoders.eu/algorithms/dijkstras-algorithm-java/

Zhuang, Y., Kang, Y., Huang, L., & Fang, Z. (2018). A Geocoding Framework for Indoor Navigation based on the QR Code. *2018 Ubiquitous Positioning, Indoor Navigation and Location-Based Services (UPINLBS)*, 1–4. https://doi.org/10.1109/UPINLBS.2018.8559741

ANEXOS

ANEXOS

Anexo 1. Historias de Usuario.

	Historia de Usuario
Numero: 1	**Usuario:** Cliente
Nombre historia: Selección de marcado en el piso	
Prioridad: Alta	**Riesgo en desarrollo:** Medio **(Alto / Medio / Bajo)**
Puntos estimados: 3	**Iteración asignada:** 1
Programador Responsable: Ronny Calle	
Descripción: Se debe buscar un sistema de marcado de piso para ser usado en el aplicativo móvil	
Observaciones: Debe permitir ser reconocible en cualquier entorno o superficie	

	Historia de Usuario
Numero: 2	**Usuario:** Cliente
Nombre historia: Mapa de interiores	
Prioridad: Alta	**Riesgo en desarrollo:** Medio **(Alto / Medio / Bajo)**
Puntos estimados: 3	**Iteración asignada:** 2
Programador Responsable: Ronny Calle	
Descripción: Se debe mostrar el mapa de interiores tanto en el aplicativo móvil y web	
Observaciones:	

	Historia de Usuario
Numero: 3	**Usuario:** Cliente
Nombre historia: Lectura del sistema de marcado en el piso	
Prioridad: Media	**Riesgo en desarrollo:** Medio **(Alto / Medio / Bajo)**
Puntos estimados: 2	**Iteración asignada:** 3
Programador Responsable: Ronny Calle	
Descripción: El aplicativo móvil debe permitir reconocer el sistema de marcado de piso seleccionado previamente de la iteración 1 y ubicar esa posición dentro del mapa de interiores.	
Observaciones:	

	Historia de Usuario
Numero: 4	**Usuario:** Cliente
Nombre historia: Visualización de los marcadores en el aplicativo	
Prioridad: Alta	**Riesgo en desarrollo:** Medio **(Alto / Medio / Bajo)**

Puntos estimados: 3	**Iteración asignada:** 4
Programador Responsable: Ronny Calle	
Descripción: El aplicativo móvil debe mostrar la localización de punto de ida y así como los marcadores destinos al que puede mostrar la ruta corta	
Observaciones:	

	Historia de Usuario
Numero: 5	**Usuario:** Cliente
Nombre historia: Ruta corta	
Prioridad: Alta	**Riesgo en desarrollo:** Alto **(Alto / Medio / Bajo)**
Puntos estimados: 4	**Iteración asignada:** 5
Programador Responsable: Ronny Calle	
Descripción: El aplicativo móvil de mostrar la ruta corta desde el punto de inicio hacia al punto de llegada	
Observaciones:	

	Historia de Usuario
Numero: 6	**Usuario:** Cliente
Nombre historia: Ingreso de nuevos marcadores para el mapa de interiores	
Prioridad: Media	**Riesgo en desarrollo:** Medio **(Alto / Medio / Bajo)**
Puntos estimados: 3	**Iteración asignada:** 6
Programador Responsable: Ronny Calle	
Descripción: El aplicativo web debe permitir ingresar nuevos marcadores que funcionaran en el aplicativo móvil	
Observaciones: Debe permitir ubicar el marcador en el mapa de interiores de manera fácil	

	Historia de Usuario
Numero: 7	**Usuario:** Cliente
Nombre historia: Eliminación de marcadores para el mapa de interiores	
Prioridad: Baja	**Riesgo en desarrollo:** Medio **(Alto / Medio / Bajo)**
Puntos estimados: 3	**Iteración asignada: 7**
Programador Responsable: Ronny Calle	
Descripción: El aplicativo web debe permitir eliminar marcadores para que ya no se muestren en el aplicativo móvil	
Observaciones:	

	Historia de Usuario
Numero: 8	**Usuario:** Cliente
Nombre historia: Actualización de marcadores para el mapa de interiores	
Prioridad: Baja	**Riesgo en desarrollo:** Medio **(Alto / Medio / Bajo)**
Puntos estimados: 3	**Iteración asignada: 8**

Programador Responsable: Ronny Calle	
Descripción: El aplicativo web debe permitir actualizar marcadores para cambiar su localización dentro del mapa de interiores	
Observaciones:	
	Historia de Usuario
Numero: 9	**Usuario:** Cliente
Nombre historia: Generar código QR	
Prioridad: Baja	**Riesgo en desarrollo: Bajo (Alto / Medio / Bajo)**
Puntos estimados: 3	**Iteración asignada: 9**
Programador Responsable: Ronny Calle	
Descripción: El aplicativo web debe generar el código QR para el marcador	
Observaciones:	

Anexo 2. Código fuente de la aplicación móvil.

https://github.com/ronstreet/AplicativoAndroid

Anexo 3. Código fuente página Web.

https://github.com/ronstreet/PaginaWeb