



UNIVERSIDAD DE QUINTANA ROO
DIVISIÓN DE CIENCIAS, INGENIERÍA Y TECNOLOGÍA

**DESARROLLO DE UNA APLICACIÓN
MÓVIL PARA LAS UNIDADES DE SALUD
DEL ESTADO DE QUINTANA ROO**

TESIS
PARA OBTENER EL GRADO DE
INGENIERO EN REDES

PRESENTA

JOSÉ EDUARDO UCAN ORDOÑEZ

DIRECTORA DE TESIS
MTI. MELISSA BLANQUETO ESTRADA

ASESORES

MTI VLADIMIR VENIAMIN CABAÑAS VICTOR
MTI JAIME SILVERIO ORTEGON AGUILAR
MTI RUBÉN ENRIQUE GONZÁLEZ ELIXAVIDE
DR. JAVIER VAZQUEZ CASTILLO



CHETUMAL QUINTANA ROO, MÉXICO, JUNIO DE 2021



UNIVERSIDAD DE QUINTANA ROO
DIVISIÓN DE CIENCIAS, INGENIERÍA Y TECNOLOGÍA

TRABAJO DE TESIS TITULADO
“DESARROLLO DE UNA APLICACIÓN MÓVIL PARA
LAS UNIDADES DE SALUD DEL ESTADO DE QUINTANA ROO”

ELABORADO POR

JOSÉ EDUARDO UCAN ORDOÑEZ

BAJO SUPERVISIÓN DEL COMITÉ DEL PROGRAMA DE LICENCIATURA
Y APROBADO COMO REQUISITO PARCIAL PARA OBTENER EL GRADO DE:
INGENIERO EN REDES

COMITÉ SUPERVISOR

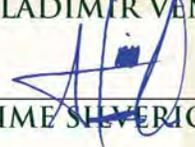
DIRECTORA:


MTI. MELISSA BLANQUETO ESTRADA

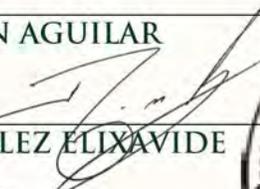
ASESOR:


MTI. VLADIMIR VENIAMÍN CABANAS VICTORIA

ASESOR:


DR. JAIME SILVERIO ORTEGÓN AGUILAR

ASESOR:


MSI. RUBÉN ENRIQUE GONZÁLEZ ELIXAVIDE

ASESOR:


DR. JAVIER VÁZQUEZ CASTILLO



CHETUMAL QUINTANA ROO, MÉXICO, JUNIO DE 2021

Resumen

El proyecto realizado en este trabajo de titulación tiene como finalidad acercar a la población en general a los Servicios Estatales de Salud (SESA) en Quintana Roo a través del desarrollo de una aplicación móvil. Entre las principales características de la aplicación desarrollada se encuentra el listado de unidades médicas, de acuerdo con la ubicación del usuario, brindando información de contacto, especialidades y servicios. El usuario cuenta con la facilidad de explorar el catálogo de unidades médicas de SESA, filtrar la búsqueda y determinar cuál de ellas se adapta a sus necesidades y de ser el caso, agendar una cita médica.

Un mapa es una forma fácil de orientar al usuario en el entorno que le rodea, una característica fundamental de la aplicación móvil es su integración con la *API (Application Programming Interface)* de *Google Maps*, esta librería permite determinar la localidad en la que se encuentra el usuario y brinda las herramientas necesarias para disponer de un mapa dinámico en el que se marcan las unidades médicas más cercanas a su posición geográfica actual de su localidad, posicionamiento que se puede seguir en todo momento en este mismo mapa, así como trazar la mejor ruta hacia cualquier unidad de salud.

La aplicación móvil se desarrolló con *ionic* en su versión 4, el cual se basa en lenguajes para programación web, es decir HTML, CSS y JS. Este framework combina la tecnología de Angular JS y Apache Córdoba para brindar una aplicación móvil con soporte multiplataforma.

Agradecimientos

A mis padres, por el apoyo y amor incondicional que siempre me han brindado, por hacer de mí una persona de bien, inculcándome valores y principios, gracias por apoyarme en todo y siempre estar ahí para mí.

A mi hermana, que ha sido mi confidente y mejor amiga, gracias por todo lo que siempre has hecho por mí, por darme todo tu cariño y ayudarme a lograr todas mis metas.

A mis profesores por compartirme sus experiencias y conocimiento, guiarme y asesorarme a lo largo de todo mi recorrido universitario, gracias por siempre estar dispuestos a escucharme y aconsejarme.

A mi colega y amigo Jafet Osorio, por formar parte fundamental de este proyecto, por todas las horas programando juntos, gracias por compartirme tus conocimientos, siempre me has ayudado a buscar soluciones y alternativas a los problemas que se me presentan.

A mis amigos, estos últimos años tuve la fortuna de hacer crecer ese círculo de amistad con grandes personas que conocí en la Universidad, hemos compartido grandes momentos dentro y fuera del aula, siempre nos apoyábamos los unos a otros para resolver nuestros problemas, gracias por el apoyo y todos los buenos momentos que pasamos juntos.

Dedicatoria

El siguiente trabajo está dedicado a mi familia por ser la fuente de mi inspiración para superarme cada día y ser el pilar más importante para seguir el camino del bien, por su amor, trabajo y apoyo que me han brindado siempre.

A la maestra Melissa Blanqueto Estrada quien fungió como mi asesora de tesis, por guiarme pacientemente en el desarrollo de este trabajo de tesis, por ayudarme a resolver todas mis dudas, brindarme consejos y siempre hacer un espacio para recibirme.

Al L.I. Sergio Alcocer Betancourt por darme la oportunidad de formar parte de este proyecto y facilitarme las herramientas necesarias para su realización.

Finalmente, se lo dedico también a todas las personas que de alguna u otra forma me apoyaron para que este proyecto sea posible, que con un simple gesto o acción lograron un cambio positivo.

Contenido

Resumen	i
Agradecimientos	ii
Dedicatoria	iii
Índice de figuras	vi
Índice de tablas	viii
Capítulo 1 Introducción.....	1
1.1 Antecedentes	1
1.2 Justificación	2
1.2.1 Importancia y facilidad de acceso a la tecnología móvil	3
1.3 Objetivo general	4
1.4 Objetivos particulares.....	5
1.5 Alcance	5
Capítulo 2 Marco Teórico	6
2.1 La tecnología móvil	6
2.1.1 Desarrollo de aplicaciones híbridas	7
2.2 Metodología para el desarrollo de aplicaciones móviles	7
2.2.1 Metodología XP (<i>Extreme Programming</i>).....	8
2.3 Tecnología para la construcción del software	10
Capítulo 3 Desarrollo.....	14
3.1 Fase de planificación	14
3.1.1 Descripción de los interesados	14
3.1.2 Requerimientos funcionales	15
3.1.3 Requerimientos no funcionales	16
3.1.4 Restricciones	17
3.2 Fase de diseño	18
3.2.1 Diagrama arquitectónico	18

3.2.2 Diagrama de navegación	19
3.2.3 Modelo de datos.....	20
3.2.4 Diseño de las interfaces gráficas	25
3.3 Fase de Codificación.....	27
3.3.1 Desarrollo de la pantalla de bienvenida	27
3.3.2 Desarrollo de la pantalla “Unidades”	31
3.3.3 Desarrollo de la funcionalidad de filtros.....	37
3.3.4 Desarrollo de la pantalla “Detalles”	38
3.3.5 Desarrollo de la pantalla “Mapa”	44
3.3.6 Desarrollo de la funcionalidad de “Citas”.....	53
3.4 Fase de Pruebas.....	60
3.4.1 Elaboración del plan de pruebas	60
3.4.2 Ejecución de las pruebas	64
3.5 Documentación	65
Capítulo 4 Conclusiones.....	66
4.1 Trabajo a futuro	67
Referencias	68
Anexos	70
Anexo 1. Configuración del entorno de desarrollo	70
Descarga del proyecto	70
Instalación de dependencias.....	71
Despliegue de la aplicación móvil	73
Anexo 2. Infografía de la aplicación móvil.....	79
Anexo 3. Organigrama Servicios Estatales de Salud	82
Anexo 4. Modelado de datos	83
Anexo 5. Ejecución de pruebas.....	84

Índice de figuras

Figura 1. Flujo para recibir algún servicio médico en las unidades de salud de SESA.....	1
Figura 2. Mapa de jurisdicciones en Quintana Roo (SESA, 2019)	2
Figura 3. Usuarios de Internet según equipo de conexión (ENDUTIH, 2018).....	4
Figura 4. Proceso de desarrollo de software con metodologías ágiles.....	7
Figura 5. Ciclo de la metodología Extreme Programming (Joskowicz, 2008).....	8
Figura 6. Ejemplo de etiquetas Ionic. (Ionic, 2020).....	11
Figura 7. Ejemplo básico data-binding con angular. (ANGULAR JS, 2020)	12
Figura 8. Integración de Cordova en dispositivos móviles (Griffith, 2017).	13
Figura 9. Diagrama de despliegue de la aplicación móvil.	18
Figura 10. Diagrama de navegación de la aplicación móvil.	19
Figura 11. Modelo de datos de la aplicación móvil.	20
Figura 12. Diagrama relacional de las unidades de salud de SESA.....	21
Figura 13. Diagrama relacional especialidades y servicios de una unidad de salud.	22
Figura 14. Diagrama relacional citas médicas.	22
Figura 15. Diagrama relacional de Laravel Passport.	23
Figura 16. Diagrama relacional de Spatie Permission.	24
Figura 17. Pantalla de bienvenida, listado de unidades y mapa de la aplicación móvil.	25
Figura 18. Agenda médica, detalles de la unidad de salud y filtros de búsqueda.....	26
Figura 19. Proceso de autenticación con Laravel Passport.....	27
Figura 20. Error de conexión con el servidor.	28
Figura 21. Uso básico del plugin de geolocalización Apache Cordova. (Ionic, 2020).....	29
Figura 22. Reverse Geocoding filtrado por tipo. (Google, 2020).....	30
Figura 23. Resultado de la pantalla de bienvenida de la aplicación móvil.	30
Figura 24. Estructura de respuesta con el listado de unidades.	31
Figura 25. Componente Infinite Scroll de Ionic.	32
Figura 26. Ion-Refresher para Android y iOS. (Ionic, 2020).....	33
Figura 27. Resultado del componente Ion Refresher en la aplicación móvil.....	34
Figura 28. Pantalla unidades de la aplicación móvil.	35
Figura 29. Ejemplo de uso del plugin "Launch Navigator".	35
Figura 30. Ejemplo de ruta hacia una unidad de salud con Google Maps.	36
Figura 31. Pantalla de búsqueda de unidades de salud.	37

Figura 32. Variable que contiene los filtros seleccionados del usuario.....	37
Figura 33. Filtrado de unidades por tipo de unidad.....	38
Figura 34. Respuesta de la solicitud con la información detallada de la unidad.....	39
Figura 35. Pantalla de detalles de la aplicación móvil y menú de acciones.....	40
Figura 36. Sección de especialidades en la pantalla de detalles.....	41
Figura 37. Modal para especialidades médicas.....	42
Figura 38. Modal de personal médico que imparte una especialidad o servicio.....	42
Figura 39. Sección de servicios en la pantalla de detalles.....	43
Figura 40. Modal para servicios médicos.....	43
Figura 41. Tarifas para todas las APIs de Mapas. (Google, 2020).....	44
Figura 42. Tarifas para todas las APIs de Rutas. (Google, 2020).....	45
Figura 43. Fracción de tarifas para las API de Places.....	46
Figura 44. Conjunto predeterminado de controles para un mapa nuevo.....	47
Figura 45. Configuración de controles para el mapa de la pantalla "Mapa".....	49
Figura 46. Pantalla de "Mapa", vista de tarjetas de información minimizada.....	50
Figura 47. Pantalla de "Mapa", vista de tarjetas de información maximizada.....	51
Figura 48. Ejemplo para crear un marcador con la API de Google Maps. (Google, 2020).....	52
Figura 49. Botón para iniciar el rastreo y mostrar el marcador de posicionamiento.....	53
Figura 50. Botones para la funcionalidad de citas médicas.....	53
Figura 51. Inicio de sesión para la funcionalidad de citas médicas.....	54
Figura 52. Formulario de registro para la funcionalidad de citas médicas.....	54
Figura 53. Información personal obtenida de RENAPO.....	55
Figura 54. Posibles mensajes de alerta al crear una cuenta en la aplicación móvil.....	55
Figura 55. Calendario para consultar la agenda médica de una unidad de salud.....	56
Figura 56. Alertas para un usuario que ya cuenta con una cita agendada.....	56
Figura 57. Posibles respuestas al consultar la agenda médica de una unidad.....	57
Figura 58. Mensaje de confirmación para agendar una cita médica.....	58
Figura 59. Solicitud para reestablecer contraseña.....	59
Figura 60. Notificación de correo enviado exitosamente para reestablecer contraseña....	59
Figura 61. Clonación del proyecto.....	70
Figura 62. Instalación de dependencias del proyecto.....	71
Figura 63. Instalación de Ionic CLI.....	72
Figura 64. Aplicación móvil corriendo en un entorno local de desarrollo.....	72

Figura 65. Instalación de la herramienta de línea de comandos Android SDK.	73
Figura 66. Agregar Gradle a las variables del sistema.	74
Figura 67. Agregar Gradle a la variable path del sistema.	75
Figura 68. Verificación de instalación de Gradle.	75
Figura 69. APK sin firmar de la aplicación móvil.	76
Figura 70. comando para generar una llave privada.	77
Figura 71. Llave generada para la aplicación móvil.	77
Figura 72. Comando para firmar una APK.	78
Figura 73. Infografía de la aplicación móvil. Parte 1 de 3	79
Figura 74. Infografía de la aplicación móvil. Parte 2 de 3	80
Figura 75. Infografía de la aplicación móvil. Parte 3 de 3	81
Figura 76. Organigrama SESA (SESA, 2020)	82
Figura 77 Modelo de datos de la aplicación móvil.	83

Índice de tablas

Tabla 1. Listado de pruebas unitarias.	61
Tabla 2. Pruebas de integración con el back-end de la aplicación móvil.	62
Tabla 3. Pruebas de integración con el Web Service de Google Maps.	62
Tabla 4. Pruebas de integración con la API de RENAPO.	62
Tabla 5. Listado de pruebas de interfaces.	63
Tabla 6. Ejecución de la prueba listar unidades de salud.	64
Tabla 7. Ejecución de la prueba detalles de una unidad de salud.	84
Tabla 8. Ejecución de la prueba filtrar unidades de salud.	85
Tabla 9. Ejecución de la prueba mapa de unidades de salud.	86
Tabla 10. Ejecución de la prueba seguir el posicionamiento del usuario.	87
Tabla 11. Ejecución de la prueba ruta hacia una unidad de salud.	88
Tabla 12. Ejecución de la prueba registro de cuentas de usuario.	89
Tabla 13. Ejecución de la prueba restablecimiento de contraseñas.	90
Tabla 14. Ejecución de la prueba agendar una cita médica.	91

Capítulo 1 Introducción

1.1 Antecedentes

Actualmente existen diversas aplicaciones en materia de salud. En el caso de México, una de las aplicaciones en el ámbito de salud proviene del IMSS (Instituto Mexicano del Seguro Social). IMSS Digital es una aplicación móvil que ofrece un portal de acceso a servicios médicos para los derechohabientes de esta institución de salud del gobierno federal.

Desde su arranque en diciembre del 2015 al cierre de junio del 2017 la App IMSS Digital era la aplicación de salud número uno en descargas en México, alcanzando 1.3 millones de descargas en el país. En ese mismo periodo se realizaron 4.4 millones de trámites y servicios a través de la aplicación. (IMSS, 2016)

Hasta el momento no existe una aplicación móvil para brindar a la sociedad quintanarroense un portal de acceso a los Servicios Estatales de Salud similar a como lo hacen las instituciones de salud a nivel federal. Actualmente para que un usuario del sector salud estatal disponga de algún servicio médico debe seguir los siguientes pasos:

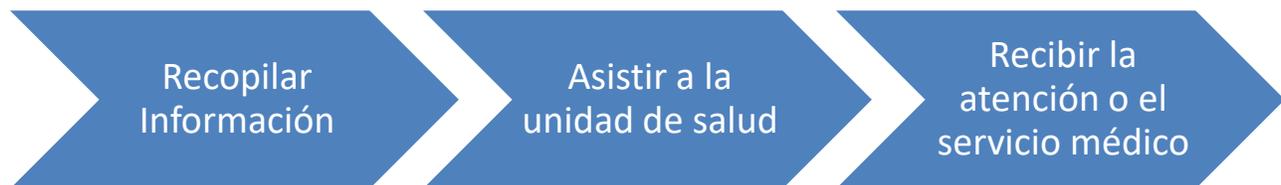


Figura 1. Flujo para recibir algún servicio médico en las unidades de salud de SESA

Recopilar información: El usuario debe recopilar información de las diferentes unidades de salud y los servicios que éstos ofrecen, generalmente esta información es consultada a través de vecinos, medios electrónicos o acudiendo a cualquier unidad de salud para solicitar más información.

Asistir a la unidad de salud: Una vez que el usuario identifica la unidad de salud debe averiguar la ruta que tiene que seguir para llegar a ella, aunque parezca una acción sencilla se puede volver compleja cuando el usuario viaja a una ciudad vecina, poblado rural o se ubica en una zona desconocida de la ciudad. Al llegar a la unidad de salud el usuario debe verificar que se le puede brindar la atención médica que necesita.

Recibir la atención o el servicio médico: Para recibir un servicio o atención médica, el usuario debe acudir desde muy temprano a la unidad para tener acceso a una ficha, deberá esperar su turno y finalmente podrá ser atendido.

1.2 Justificación

La Jurisdicción Sanitaria es una entidad de los Servicios Estatales de Salud con capacidad para la planeación, administración, dirección, operación y evaluación de los recursos del primero y segundo niveles de atención para satisfacer las necesidades de salud de la población abierta y la coordinación con los servicios correspondientes de la seguridad social. (Ruiz de Chavez & Martínez Narváez, 1988)



Figura 2. Mapa de jurisdicciones en Quintana Roo (SESA, 2019)

La Jurisdicción Sanitaria No.1, Jurisdicción Sanitaria No.2 y Jurisdicción Sanitaria No. 3 del Estado de Quintana Roo están a cargo de unidades de salud, tales como: hospitales generales, hospitales integrales, centros de salud urbanos, centros de salud rurales y unidades de especialidades médicas (UNEMEs).

De acuerdo con el Anuario Estadístico y Geográfico de Quintana Roo elaborado por el INEGI (2017), estas unidades de salud atendieron al 40.71% de la población total del Estado de Quintana Roo a través del programa federal anteriormente conocido como “Seguro Popular”. Esta cifra que forma parte del público objetivo aumenta al incluir la población que asistió a las unidades de salud para recibir un servicio médico y no estaban afiliados a dicho programa.

De acuerdo con los siguientes datos estadísticos sobre los usuarios que tienen acceso a un celular inteligente, implementar una aplicación móvil es una alternativa de fácil acceso al público objetivo y que les permitirá hacer uso de esta plataforma en cualquier momento.

1.2.1 Importancia y facilidad de acceso a la tecnología móvil

Las tecnologías móviles tienen el potencial de revolucionar la forma en la que la sociedad se relaciona con los servicios de salud, de acuerdo con el informe emitido por la Organización Mundial de la Salud (2016) “está demostrado que las tecnologías móviles inalámbricas en la salud pública, denominadas «mSalud», potencian el acceso a la información, servicios y competencias sanitarios, además de fomentar cambios positivos en los comportamientos en materia de salud para prevenir el inicio de enfermedades agudas y crónicas.”

En este mismo informe se habla sobre la facilidad del acceso a la telefonía móvil, haciendo mención que de los 7000 millones de usuarios con acceso a este servicio el 70% de ellos provienen de países de ingresos bajos o medianos, incluso, en muchos lugares, hay más probabilidades de tener acceso a un teléfono móvil que a agua limpia, una cuenta bancaria o electricidad.

Para el caso de México y de acuerdo con la Encuesta Nacional sobre Disponibilidad y Uso de Tecnologías de la Información en los Hogares (ENDUTIH, 2018) el número de usuarios que disponen de un smartphone es de 69.6 millones, con un incremento de 4.9 millones entre el año 2017 y 2018. Tomando en cuenta los dispositivos utilizados para conectarse a Internet, en 2018 la ENDUTIH dio a conocer que el 92.7% de los usuarios de Internet se conectaron a través de un celular inteligente (smartphone), ver Figura 3. ¹



Figura 3. Usuarios de Internet según equipo de conexión (ENDUTIH, 2018)

“Las tecnologías móviles se están convirtiendo en un importante recurso en la prestación de servicios de salud y la salud pública gracias a su facilidad de uso, enorme difusión y amplia aceptación”. (OMS, 2016)

1.3 Objetivo general

Brindar a los usuarios finales un portal de acceso a servicios médicos con el desarrollo de una aplicación móvil para las unidades de salud del Estado de Quintana Roo.

¹ Nota: Los usuarios pueden utilizar más de un equipo de conexión.

1.4 Objetivos particulares

- Determinar la manera en la que operan las unidades de salud del Estado de Quintana Roo al agendar citas médicas.
- Identificar las herramientas necesarias para el desarrollo y ejecución de la aplicación propuesta.
- Ubicar la unidad de salud más cercana de acuerdo con las necesidades del usuario y su ubicación geográfica actual.
- Agendar una cita en una unidad de salud desde la aplicación móvil.
- Gestionar las citas agendadas en cada unidad de salud.

1.5 Alcance

El alcance de este proyecto contempla el desarrollo de la interfaz gráfica de una aplicación móvil multiplataforma que permitirá el acceso a la ubicación e información de interés de las unidades de los Servicios Estatales de Salud en Quintana Roo, búsquedas de unidades y la obtención de la ruta hacia la unidad de salud seleccionada, así como agendar citas médicas.

Capítulo 2 Marco Teórico

2.1 La tecnología móvil

La tecnología móvil y en especial la llegada de los smartphones han revolucionado la forma en que nos comunicamos diariamente, poniendo en nuestras manos todos los servicios que nos puede ofrecer Internet y brindando mejores velocidades de transmisión con la llegada de nuevas generaciones de tecnologías de telefonía móvil.

En relación con otros sistemas de radiocomunicaciones, los sistemas de comunicaciones móviles aportan movilidad completa, es decir, permiten la comunicación con cualquier terminal que esté dentro de la zona de cobertura, y pueden mantener la comunicación mientras el terminal se desplaza, siempre que no se supere la velocidad máxima de diseño.

“Los sistemas de radiocomunicaciones móviles permiten el intercambio de información entre terminales a bordo de vehículos o transportados por personas y terminales fijos. Los sistemas móviles proporcionan comunicaciones con ubicuidad, versatilidad y flexibilidad. La comunicación del terminal móvil se realiza a través de una interfaz aire o interfaz radio, a través de la cual enlaza directamente con una estación base, estación fija que a su vez está conectada con la red fija. La cobertura de las estaciones base se ve en ocasiones suplementada mediante estaciones repetidoras, que permiten extender la cobertura superficial en determinadas direcciones o cubrir zonas de sombra, incluyendo túneles o interiores de edificios”. (Hernando Rábanos, Mendo Tomás, & Riera Salís, 2015)

2.1.1 Desarrollo de aplicaciones híbridas

El desarrollo de aplicaciones híbridas ha permitido a los desarrolladores distribuir sus aplicaciones móviles a una amplia gama de plataformas sin la necesidad de reescribir el código ya creado y tener que adaptarlo al lenguaje de programación empleado en cada plataforma móvil, ahorrando una gran cantidad de tiempo para el desarrollador al evitar la necesidad de documentarse sobre estos lenguajes de programación.

Típicamente, un desarrollador necesitará aprender y dominar el lenguaje de desarrollo específico de cada plataforma: Objective-C o Swift si está creando aplicaciones basadas en iOS, o Java si está creando aplicaciones basadas en Android.

Al aprovechar el lenguaje de programación web y algunos frameworks, los desarrolladores ahora pueden desarrollar sus aplicaciones en un mismo código base. Esto se conoce como una aplicación móvil híbrida porque combina las capacidades nativas del dispositivo móvil con la capacidad de desarrollarse utilizando tecnologías web, a diferencia de las aplicaciones móviles nativas tradicionales que se crean utilizando el lenguaje de desarrollo nativo del dispositivo. En su lugar, las aplicaciones híbridas se crean con tecnologías web (HTML, CSS y JavaScript). (Griffith, 2017)

2.2 Metodología para el desarrollo de aplicaciones móviles

Las metodologías ágiles para el desarrollo de software surgieron como una solución inmediata, garantizando la realización de proyectos en corto plazo, basada en iteraciones.

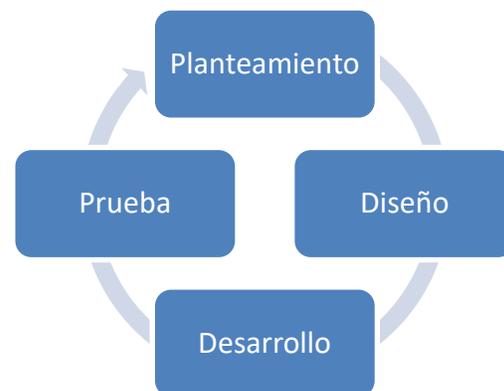


Figura 4. Proceso de desarrollo de software con metodologías ágiles.

2.2.1 Metodología XP (*Extreme Programming*)

De acuerdo con José Joskowicz (2008) *Extreme Programming* (XP) surge como una nueva manera de encarar proyectos de software, proponiendo una metodología basada esencialmente en la simplicidad y agilidad.

Se trata de realizar ciclos de desarrollo cortos (llamados iteraciones), con entregables funcionales al finalizar cada ciclo. En cada iteración se realiza un ciclo completo de análisis, diseño, desarrollo y pruebas, pero utilizando un conjunto de reglas y prácticas que caracterizan a XP.

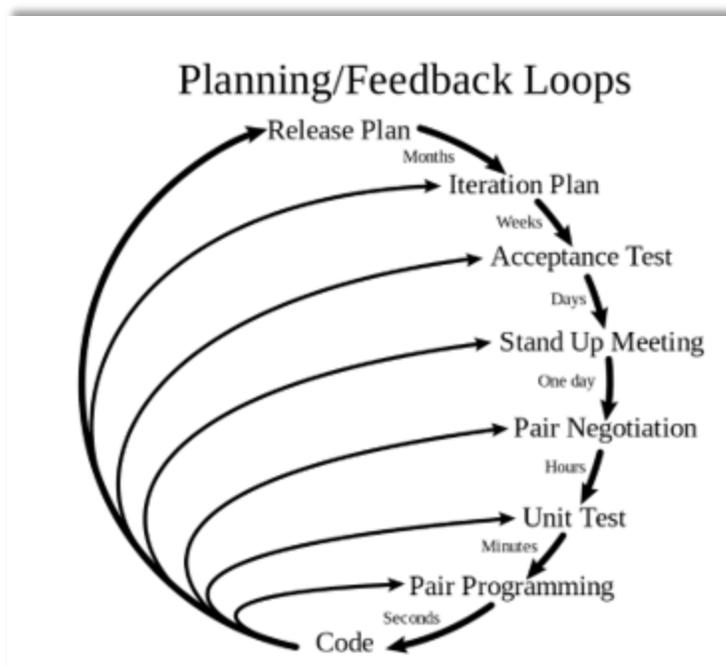


Figura 5. Ciclo de la metodología Extreme Programming (Joskowicz, 2008)

Plan de lanzamiento. El cronograma de entregas establece cuáles funcionalidades serán agrupadas para conformar una entrega, y el orden de éstas. Este cronograma será el resultado de una reunión entre todos los involucrados del proyecto.

El cronograma de entregas se realiza con base a las estimaciones de tiempos de desarrollo realizadas por los programadores. Luego de algunas iteraciones es recomendable realizar nuevamente una reunión con los involucrados del proyecto, para evaluar nuevamente el plan de entregas y ajustarlo si es necesario.

Plan de Iteración. Las funcionalidades seleccionadas para cada entrega son desarrolladas y probadas en un ciclo de iteración, de acuerdo con el orden preestablecido. Cada funcionalidad se traduce en tareas específicas de programación. Asimismo, para cada funcionalidad se establecen las pruebas de aceptación. Estas pruebas se realizan al final del ciclo en el que se desarrollan, pero también al final de cada uno de los ciclos siguientes, para verificar que subsiguientes iteraciones no han afectado a las anteriores. Las pruebas de aceptación que hayan fallado en el ciclo anterior son analizadas para evaluar su corrección, así como para prever que no vuelvan a ocurrir.

Prueba de Aceptación. Las pruebas de aceptación son creadas con base a las funcionalidades en cada ciclo de la iteración del desarrollo. El cliente debe especificar uno o diversos escenarios para comprobar que una funcionalidad ha sido correctamente implementada.

Los clientes son responsables de verificar que los resultados de estas pruebas sean correctos. Asimismo, en caso de que fallen varias pruebas, deben indicar el orden de prioridad de resolución. Una funcionalidad no se puede considerar terminada hasta tanto pase correctamente todas las pruebas de aceptación.

Reunión diaria de seguimiento (*Stand-up Meeting*). El objetivo de estas reuniones diarias es mantener la comunicación entre el equipo, compartir problemas y soluciones. En la mayoría de estas reuniones, gran parte de los participantes simplemente escuchan, sin tener mucho que aportar. Para no quitar tiempo innecesario del equipo, se sugiere realizar estas reuniones en círculo y de pie.

Negociación de pares. La negociación entre pares es muy similar a la fase de *Stand-up Meeting*, pero esta la realiza cada par de programadores y no todo el equipo, se delegan las tareas que tomará cada uno en el día, se debaten problemas y soluciones.

Pruebas unitarias. Todos los módulos deben de pasar las pruebas unitarias antes de ser liberados o publicados. Que todo código liberado pase correctamente las pruebas unitarias es lo que habilita que funcione la propiedad colectiva del código. En este sentido, el sistema y el conjunto de pruebas debe ser guardado junto con el código, para que pueda ser utilizado por otros desarrolladores, en caso de tener que corregir, cambiar o recodificar parte de este.

Programación de pares. XP propone que se desarrolle en pares de programadores, ambos trabajando juntos en una misma computadora. Si bien parece que ésta práctica duplica el tiempo asignado al proyecto, al trabajar en pares se minimizan los errores y se logran mejores diseños, compensando la inversión en horas. El producto obtenido es por lo general de mejor calidad que cuando el desarrollo se realiza por programadores individuales.

2.3 Tecnología para la construcción del software

IONIC Framework

Ionic es una combinación de varias tecnologías de código abierto que trabajan en conjunto para hacer que la creación de aplicaciones móviles sea más rápida y fácil. La capa superior de este framework es el propio *Ionic*, que proporciona la capa de interfaz de usuario de la aplicación. Justo debajo de eso está Angular JS, un framework que nos facilita el desarrollo web haciendo uso de la arquitectura MVC, ambos se ubican sobre Apache Córdoba, que permite que la aplicación web utilice las capacidades nativas del dispositivo, encargándose de renderizar el contenido web. La combinación de estas tecnologías permite a *Ionic* ofrecer una plataforma robusta para crear aplicaciones híbridas.

La característica principal del *framework Ionic* es proporcionar los componentes de la interfaz de usuario que no están disponibles para el desarrollo de aplicaciones basadas en desarrollo web. Por ejemplo, una barra de pestañas es un componente de interfaz de usuario común que se encuentra en muchas aplicaciones móviles. Pero este componente no existe como elemento web nativo, *Ionic* se encarga de recrear este componente a través de una combinación de HTML, CSS y *JavaScript*, cada componente se comporta y se parece a los controles nativos que está recreando. (Griffith, 2017)

En la Figura 6 podemos ver una lista de elementos, componente muy común en aplicaciones móviles para desglosar información en general, este componente es creado con etiquetas propias de *Ionic*, que como se mencionó anteriormente, recrean elementos nativos a través de código HTML, CSS y JS.

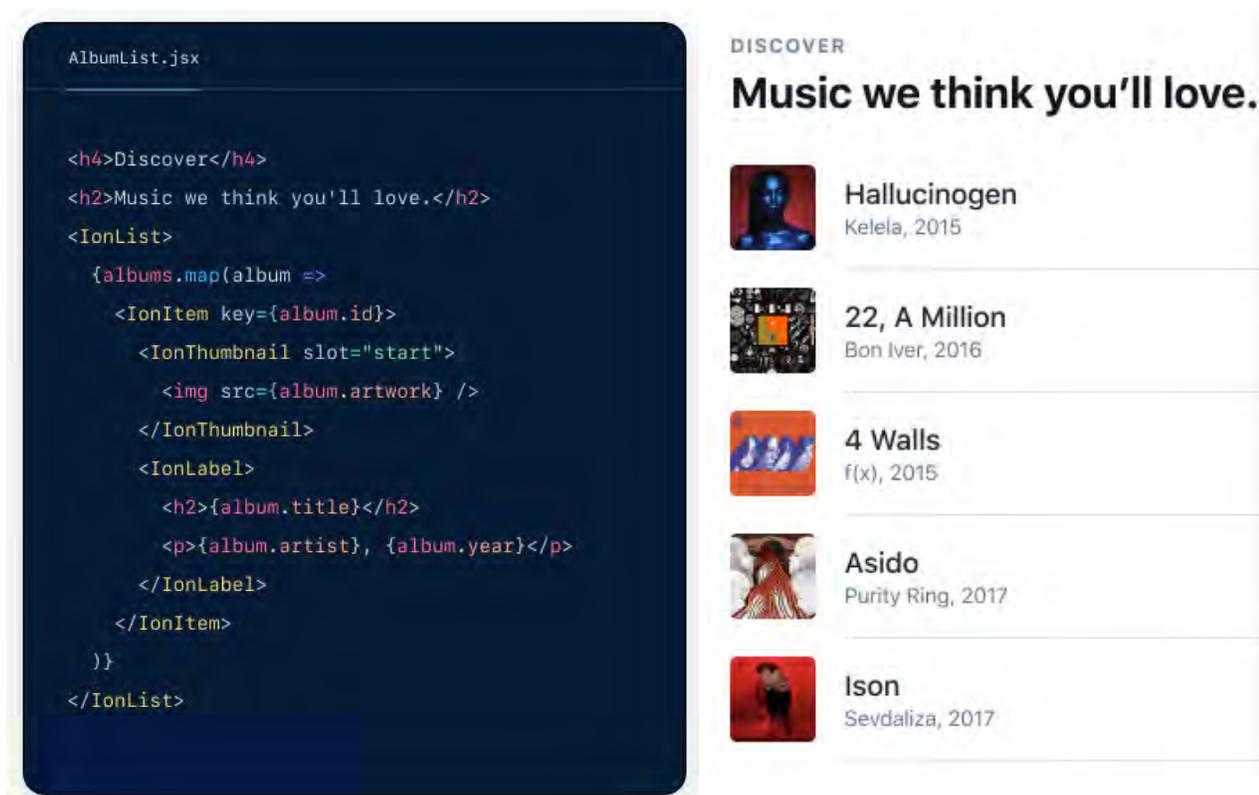


Figura 6. Ejemplo de etiquetas Ionic. (Ionic, 2020)

Más allá de los componentes de la interfaz de usuario, *Ionic Framework* se ha expandido para incluir una interfaz de línea de comandos (CLI) robusta y un conjunto de servicios adicionales como *Ionic View* e *Ionic Creator*. El enfoque principal de *Ionic* está en la capa de interfaz de usuario y su integración con Angular y Córdoba para proporcionar experiencias similares a las nativas. (Griffith, 2017)

Angular

AngularJS o Angular es un framework de JavaScript mantenido por Google para el diseño *front-end* de aplicaciones web de una sola página. Se trata de una tecnología de libre uso

que promueve y usa patrones de diseño de software, en concreto el patrón de arquitectura MVC (Modelo Vista Controlador).

La idea fundamental de Angular es mejorar el desarrollo Web de aplicaciones de una sola página y a su vez facilitar el desarrollo de pruebas sobre estos sistemas, reforzando la importancia que deben tener las pruebas en el proceso de desarrollo software. Las aplicaciones de una sola página o SPA (*Single Page Applications*) son aquellas en las que el contenido de la página se actualiza de forma dinámica, sin ser necesario una nueva carga completa de la misma, gracias al enlace de datos. De esta forma el usuario interactúa con el sistema y la aplicación se adapta automáticamente conforme a las necesidades presentes. Para ello se emplean una serie de servicios y directivas definidos por Angular, que permiten disociar la manipulación del DOM de la lógica de la aplicación. (González, 2017)

El enlace de datos (*data-binding*) proporcionado por angular es una forma automática de actualizar la vista cada vez que cambia el modelo, así como actualizar el modelo cada vez que cambia la vista, en la Figura 7 vemos un ejemplo sencillo de uso, donde se crea un input que al ingresar cualquier texto lo muestra automáticamente en pantalla gracias al enlace de datos que proporciona Angular.



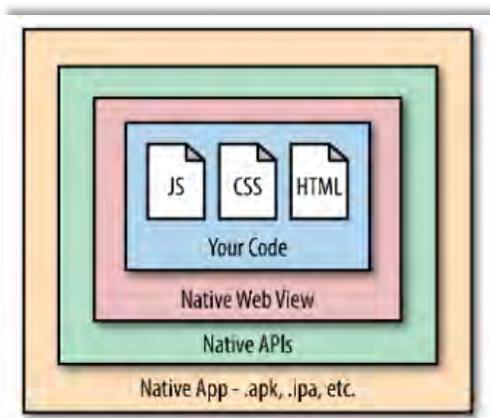
Figura 7. Ejemplo básico data-binding con angular. (ANGULAR JS, 2020)

Apache Cordova

Apache Cordova es un *framework* de desarrollo móvil gratuito y de código abierto. Permite crear aplicaciones nativas multiplataforma utilizando tecnologías web estándar. Cordova proporciona la *API* de JavaScript para acceder a las funciones nativas del dispositivo móvil.

La arquitectura de alto nivel de la aplicación Cordova contiene los siguientes elementos: Visor Web (*WebView*), componente de aplicación nativo que se utiliza para presentar contenido web en una pantalla de aplicación nativa. Aplicación Web (*WebApp*), generalmente se genera con el nombre de `index.html`, se implementa como una página web que incluye CSS, JS, iconos, imágenes y otros archivos multimedia de referencia. Además, la aplicación Cordova incluye el archivo `config.xml` que permite especificar algunos parámetros para controlar muchos aspectos del comportamiento de la aplicación. (Matrosov, Kokin, & Tyurina, 2017)

Cordova proporciona la interfaz entre el *WebView* y la capa nativa del dispositivo. La librería proporciona un *framework* para cerrar la brecha entre ambas tecnologías. Gran parte de la funcionalidad se maneja a través de un sistema de módulos de complementos (*core plugins*), que permite que la biblioteca central sea más pequeña. Más allá de trabajar en las dos plataformas móviles principales del mercado, Cordova se usa en una gama mucho más amplia de plataformas móviles, como *Windows Phone*, *BlackBerry* y *FireOS*. (Griffith, 2017)



En la Figura 8 se observan las capas que utiliza apache Cordova para funcionar como una aplicación nativa, el código que se programó con lenguaje web se renderiza en el dispositivo móvil gracias al *WebView*, este se comunica directamente con las funciones nativas del dispositivo, como resultado tenemos una aplicación que funciona igual que una nativa, que incluso se distribuye como una.

Figura 8. Integración de Cordova en dispositivos móviles (Griffith, 2017).

Capítulo 3 Desarrollo

En este capítulo se documenta cada uno de los ciclos de desarrollo que se llevaron a cabo, propuestos por la metodología *Extreme Programming* (XP). Inicialmente se describe el contenido de la aplicación móvil, que incluye todos los requerimientos esperados por los interesados (*stakeholders*); en la fase de diseño, se describe el diseño estructural y de interfaces, diseños que fueron aprobados por todos los involucrados en el proyecto; en la fase de codificación se habla del cómo se desarrolló cada módulo, las funcionalidades que incluye, las complicaciones en el desarrollo y el resultado obtenido; finalmente en la fase de pruebas se evalúa el correcto funcionamiento de cada tarea.

3.1 Fase de planificación

3.1.1 Descripción de los interesados

Stakeholders: Los interesados en el desarrollo de este proyecto están conformados por la propia Coordinación de Informática de los Servicios Estatales de Salud en Quintana Roo, haciendo cumplimiento de uno de sus objetivos que es el de proporcionar los servicios de desarrollo de sistemas, buscando una manera de brindar a los usuarios finales un portal de acceso a servicios médicos haciendo uso de las tecnologías de la información, así como acercar a la población en general a los Servicios Estatales de Salud.

Habitantes del Estado: Ciudadanos del Estado de Quintana Roo que buscan información detallada de las unidades de salud a su alrededor para hacer uso de un servicio médico, que por falta de difusión desconocen.

Turistas: Personas de otra nacionalidad o de otros estados del país, que se encuentran de visita y que buscan información detallada de las unidades de salud a su alrededor para hacer uso de un servicio médico.

3.1.2 Requerimientos funcionales

Los siguientes requerimientos funcionales surgen después de una serie de reuniones que se llevaron a cabo entre todos los involucrados en el desarrollo del proyecto, estas reuniones pertenecen al ciclo de desarrollo que lleva por nombre “plan de lanzamiento” de acuerdo con la metodología *Extreme Programming*. Estos requerimientos funcionales agrupan la idea principal de los interesados en el desarrollo de la aplicación móvil para conformar la primera versión del software.

- *Listado de unidades de salud más cercanas a la posición geográfica del usuario.*
 - Consulta de la posición geográfica del usuario.
 - Cálculo de la distancia entre el usuario y las unidades de salud.
 - Listado ordenado de la unidad de salud más cercana a la más lejana.
 - Información de interés para cada una de las unidades listadas.
 - Búsqueda de unidades de salud por localidad, tipo de unidad, especialidades y servicios.

- *Mapa dinámico para mostrar las unidades de salud en la localidad del usuario.*
 - Obtener la localidad en la que se encuentra el usuario.
 - Marcar en el mapa la ubicación de todas las unidades de salud que se encuentran en la localidad.
 - Acompañar cada uno de estos marcadores con información resumida de la unidad de salud.
 - Mostrar en todo momento en el mapa la posición geográfica actual del usuario.

- *Agenda de citas médicas.*
 - Sistema de cuentas de usuario para el uso de citas médicas.
 - Registro, inicio de sesión, restablecimiento de contraseña y verificación de cuenta.
 - Mostrar disponibilidad de agenda por unidad médica.
 - Gestión de citas generadas desde la aplicación móvil.

En este mismo ciclo de desarrollo se determinó que funcionalidades se agruparían para conformar una entrega, y el orden de éstas, cada una de estas entregas son desarrolladas y probadas en un ciclo de iteración que lleva por nombre “plan de iteración” de acuerdo con la metodología XP.

En total se obtuvieron 6 entregables, esto quiere decir que fueron necesarias 6 iteraciones para completar el desarrollo de la aplicación móvil. El plan de iteración quedo de la siguiente manera:

- Desarrollo de la pantalla de bienvenida.
- Desarrollo de la pantalla “Unidades”.
- Desarrollo de la funcionalidad de filtros.
- Desarrollo de la pantalla “Detalles”.
- Desarrollo de la pantalla “Mapa”.
- Desarrollo de la funcionalidad de “Citas”.

3.1.3 Requerimientos no funcionales

Aunque la aplicación móvil hace uso de la geolocalización para muchas funciones, ésta deberá contemplar la posibilidad de que el usuario desactive su GPS. Sin embargo, deberá seguir brindando información sobre las unidades de salud. La aplicación deberá adaptarse a los cambios de estado de este sensor, incluso monitorear el estado de la red para notificarle al usuario en caso de que se pierda la conexión con el servidor. La aplicación deberá ser capaz de manejar todos estos errores, tomar acción, notificarle de manera amigable al usuario y brindarle una solución o alternativa.

3.1.4 Restricciones

Financiero: El lanzamiento de la aplicación móvil se puede ver retrasado si el recurso no es liberado en tiempo y forma. Para que la aplicación se encuentre disponible en dispositivos Android es necesario pagar una cuota única como desarrollador en la *Google Play Store*, las cuentas de tipo desarrollador son las únicas permitidas para subir aplicaciones que se distribuyen en esta plataforma.

En el caso de la plataforma iOS, los requerimientos económicos son mayores, se necesita una computadora de la marca Apple para crear el archivo de aplicación compatible con dispositivos iOS. Para que la aplicación se encuentre en la App Store se debe pagar una cuota como desarrollador, pero en este caso se trata de una cuota anual. Finalmente, se debe de firmar un contrato con un partner de Google. Esta empresa brindará el acceso para hacer uso de la *API* de *Google Maps* y se encargará de facturar cada mes el consumo de este servicio.

Legales: Aunque la aplicación móvil tenga la funcionalidad para poder agendar una cita, esta característica debe de ser aprobada y regulada por la Dirección de Normatividad y Asuntos Jurídicos de los Servicios Estatales de Salud. La Dirección de Administración será la encargada de dar seguimiento y firmar el contrato con un *partner* de Google, si el área toma la decisión de no firmar el contrato la aplicación móvil no podrá hacer uso de la *API* de *Google Maps*.

Recopilación de la información: Toda la información relacionada con las unidades de salud que es mostrada en la aplicación móvil debe ser recopilada y dada de alta en el sistema por la Dirección de Servicios de Salud. Sin embargo, la aplicación móvil no contará con la información actualizada hasta que el área registre la información completa de cada unidad de salud.

3.2 Fase de diseño

3.2.1 Diagrama arquitectónico

Como se puede ver en la Figura 9 la aplicación móvil se conecta a tres *APIs* para funcionar, la principal de ellas es la que funge como *back-end* de la aplicación. La aplicación también consulta información del Registro Nacional de Población (RENAPO) para la obtención de información de ciudadanos a través de una CURP, información empleada en el registro de cuentas de usuario para citas médicas. Finalmente, la aplicación móvil hace uso de la *API* de *Google Maps* para hacer uso de mapas dinámicos y obtener la localidad del usuario haciendo posible mostrar las unidades médicas de su localidad. Todas las conexiones a través de internet hacia las *APIs* son por tráfico web y viajan de manera segura gracias al Protocolo Seguro de Transferencia de Hipertexto (HTTPS).

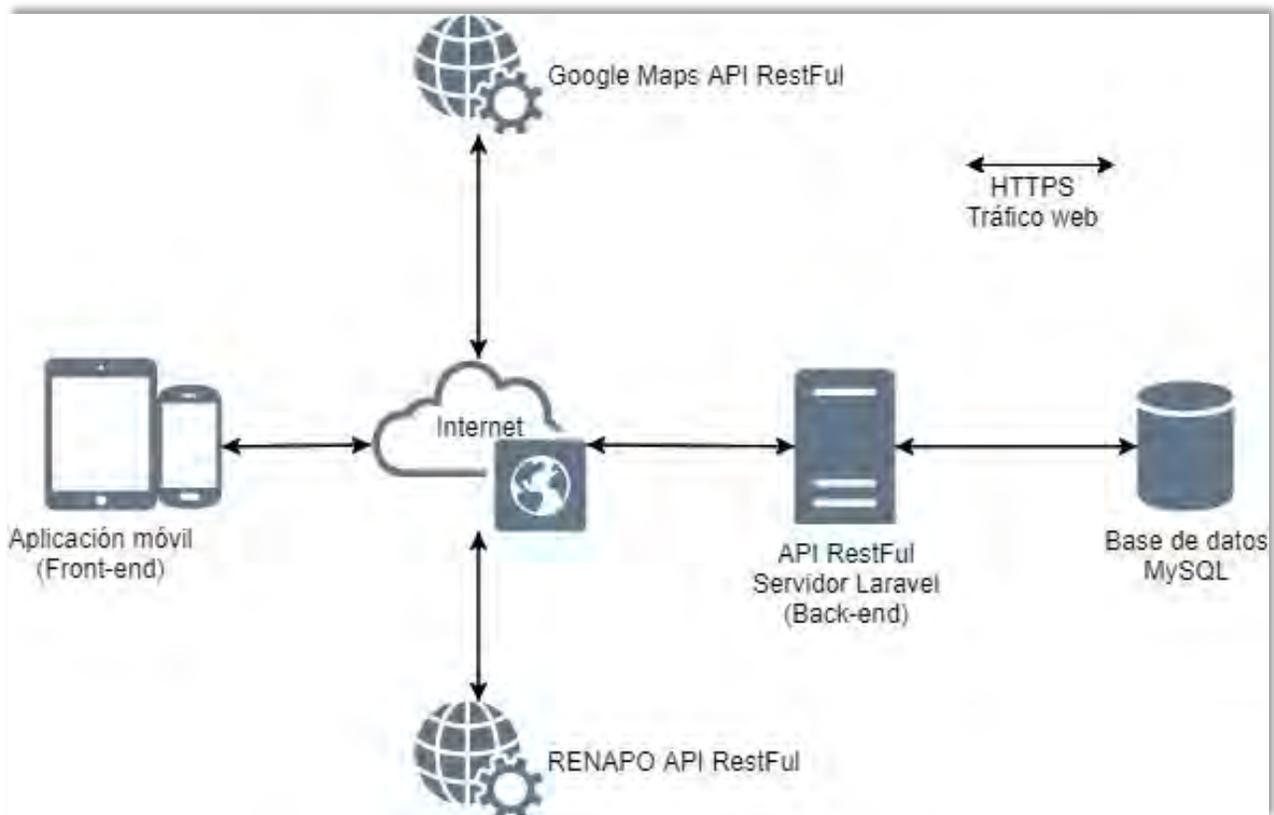
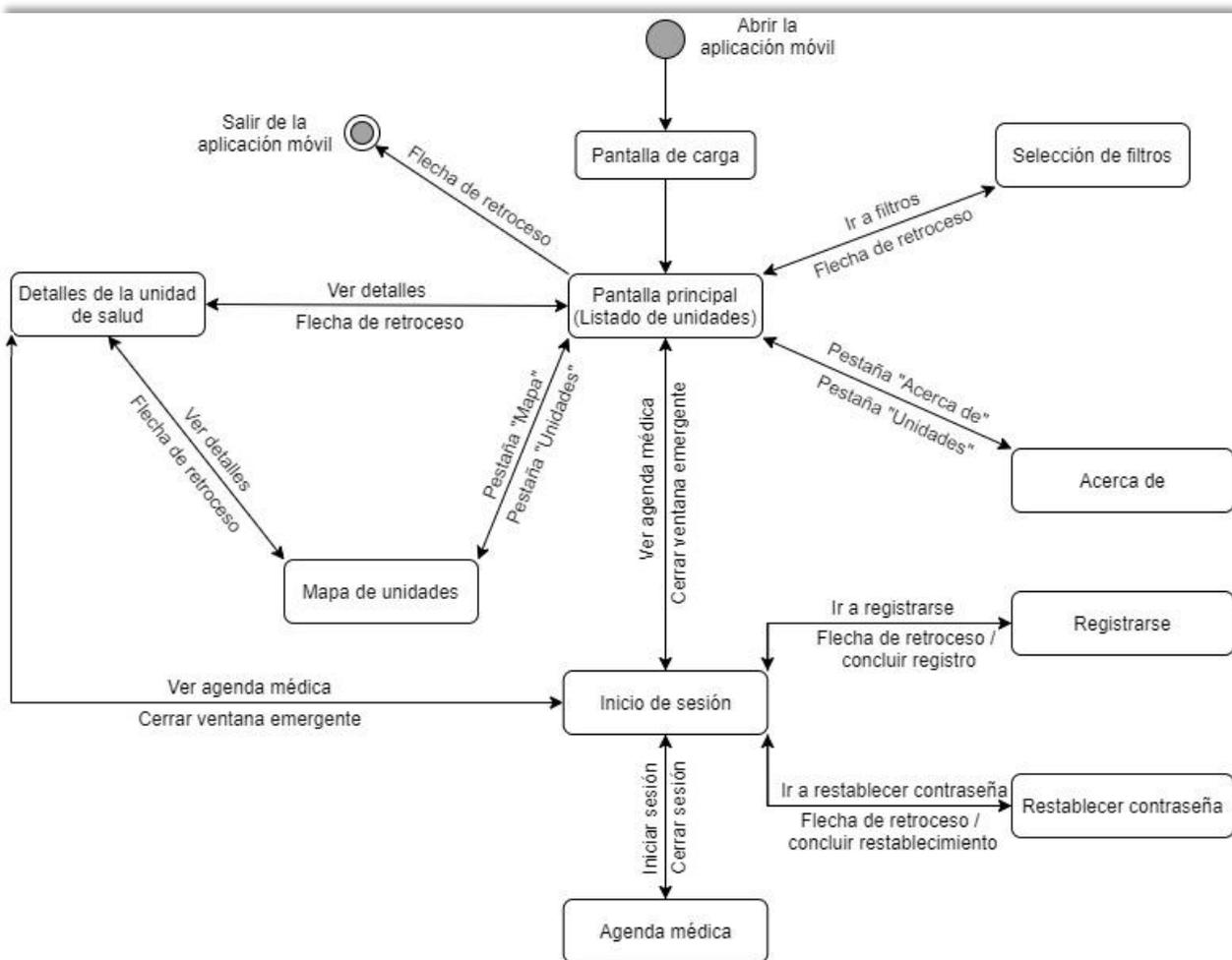


Figura 9. Diagrama de despliegue de la aplicación móvil.

3.2.2 Diagrama de navegación

La aplicación móvil implementa una navegación basada en pestañas, la pestaña principal “unidades” y dos pestañas adicionales, “mapa” y “acerca de”, cada una de estas pestañas muestra una pantalla dentro de la aplicación. Además de las pantallas anteriores la aplicación móvil cuenta con dos pantallas extras, la pantalla “detalles” y “filtros” así como una ventana emergente que controla toda la funcionalidad de citas médicas y el registro de cuentas de usuario. Ver la Figura 10.



En la Figura 12, a través de un diagrama relacional, se muestra la estructura empleada para almacenar toda la información detallada, así como fotografías de cada unidad de salud.

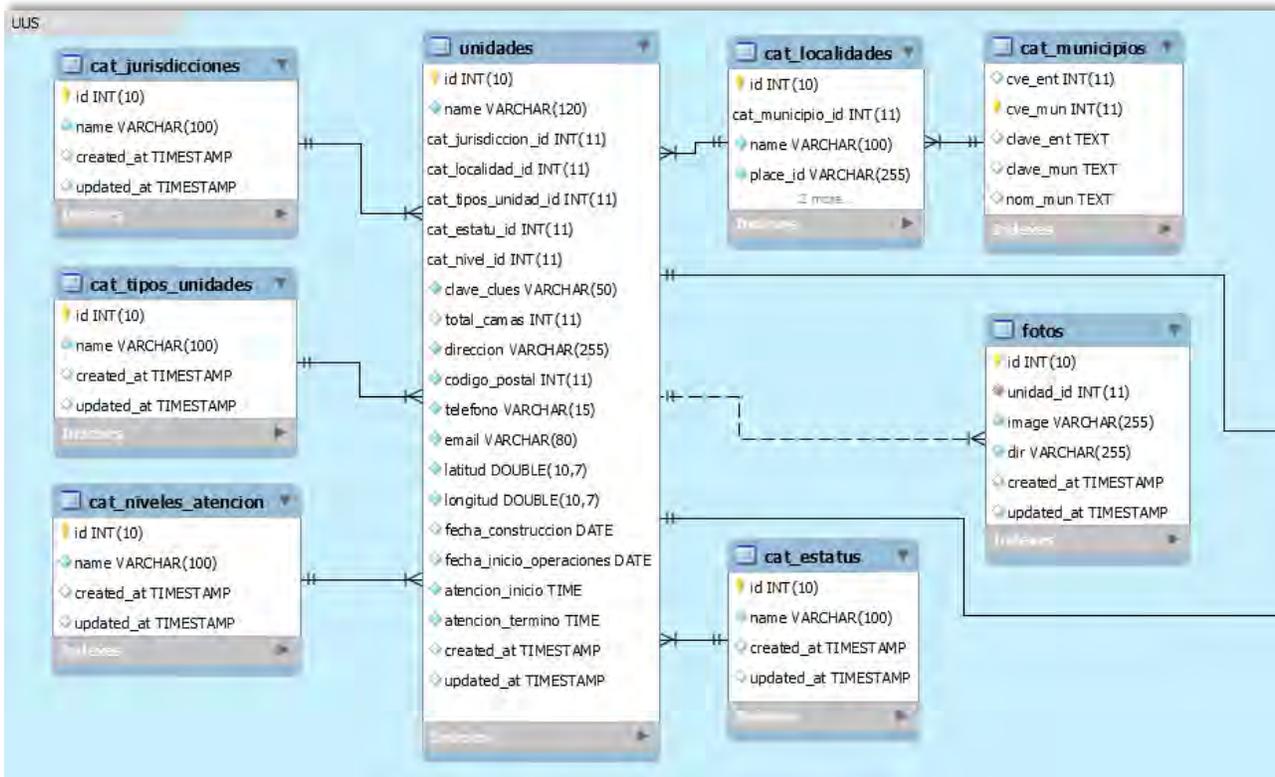


Figura 12. Diagrama relacional de las unidades de salud de SESA

Como se observa en la Figura 13 la tabla de unidades está directamente relacionada a dos tablas para almacenar la información referente a las especialidades y servicios con los que cuenta la unidad de salud, estas tablas a su vez mantienen relación con otras tablas para almacenar información de horarios de atención por cada especialidad o servicio, así como la información detallada del personal médico que imparte en cada horario de atención.

En la Figura 14 se muestran las tablas que almacenan la información de citas médicas que es generada desde la aplicación móvil, la tabla de citas almacena la fecha y hora agendada, el ciudadano que agendó, el estatus de la cita y el horario de atención del servicio en el que agendo.

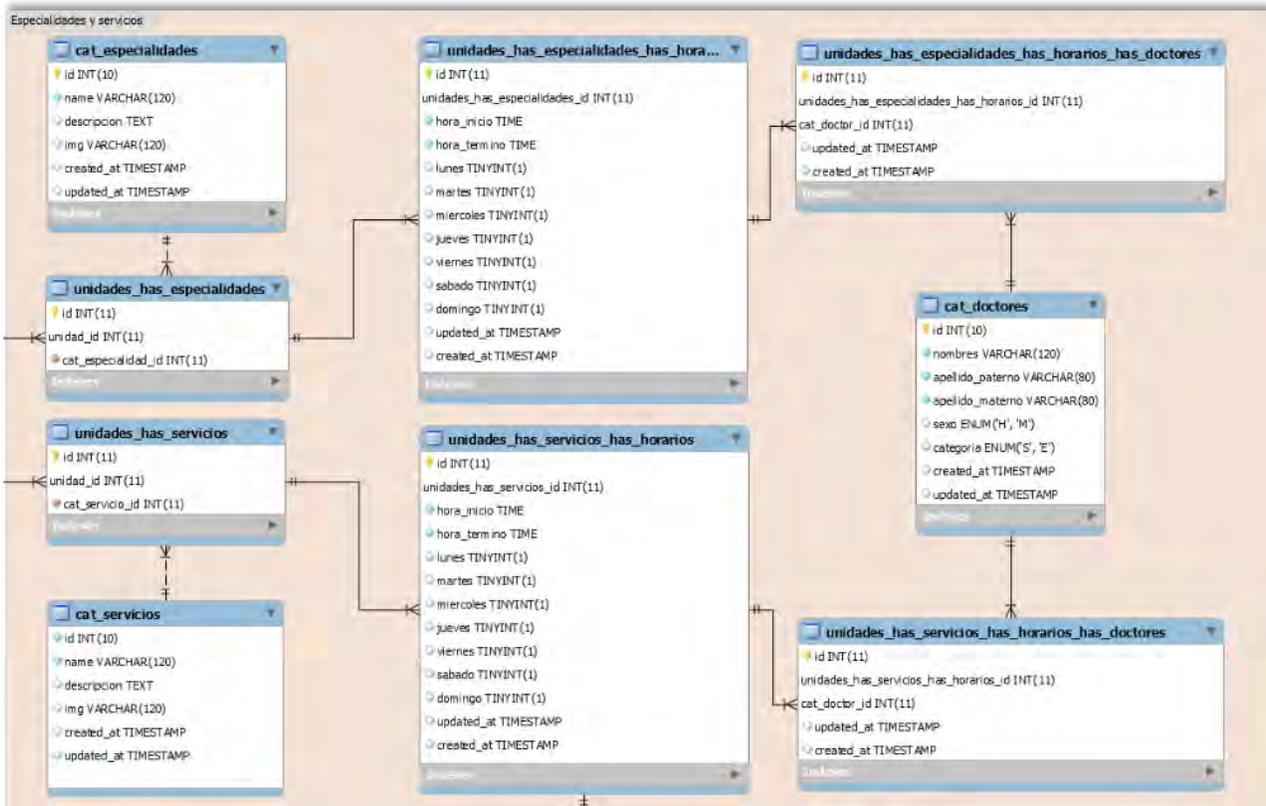


Figura 13. Diagrama relacional especialidades y servicios de una unidad de salud.

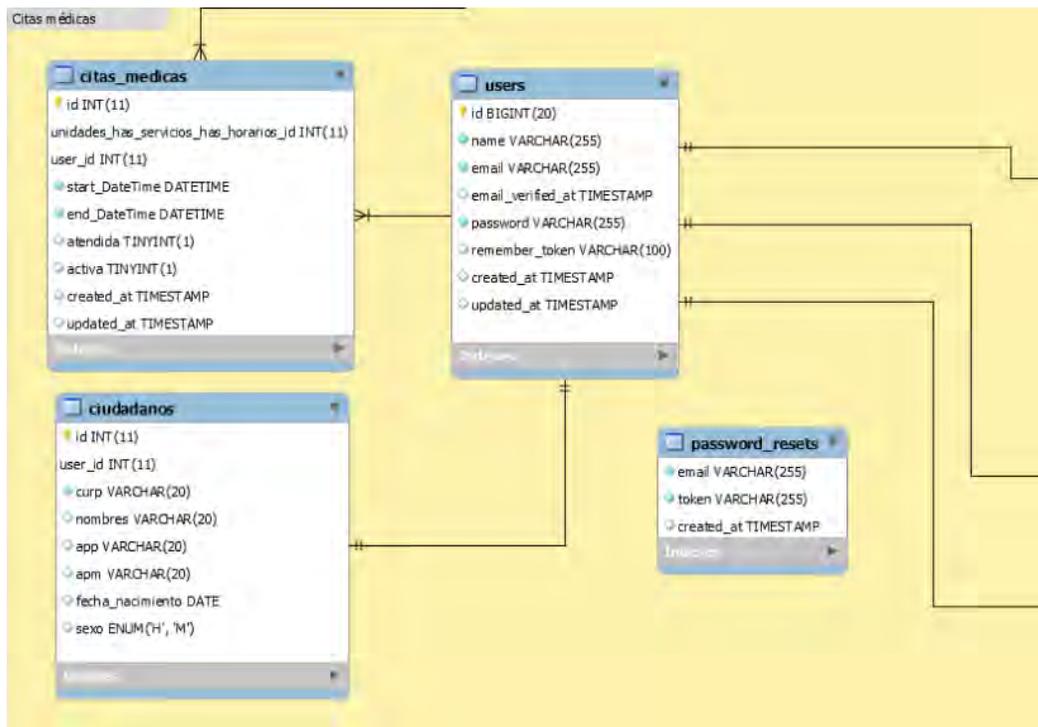


Figura 14. Diagrama relacional citas médicas.

Además de las tablas que emplea Laravel para almacenar la información de las unidades de salud se requieren tablas adicionales para funciones específicas que realiza el servidor. *Laravel Passport* genera 5 tablas requeridas para la gestión de tokens utilizados en la autenticación de usuarios en el consumo de las APIs que ofrece el servidor (ver Figura 15).

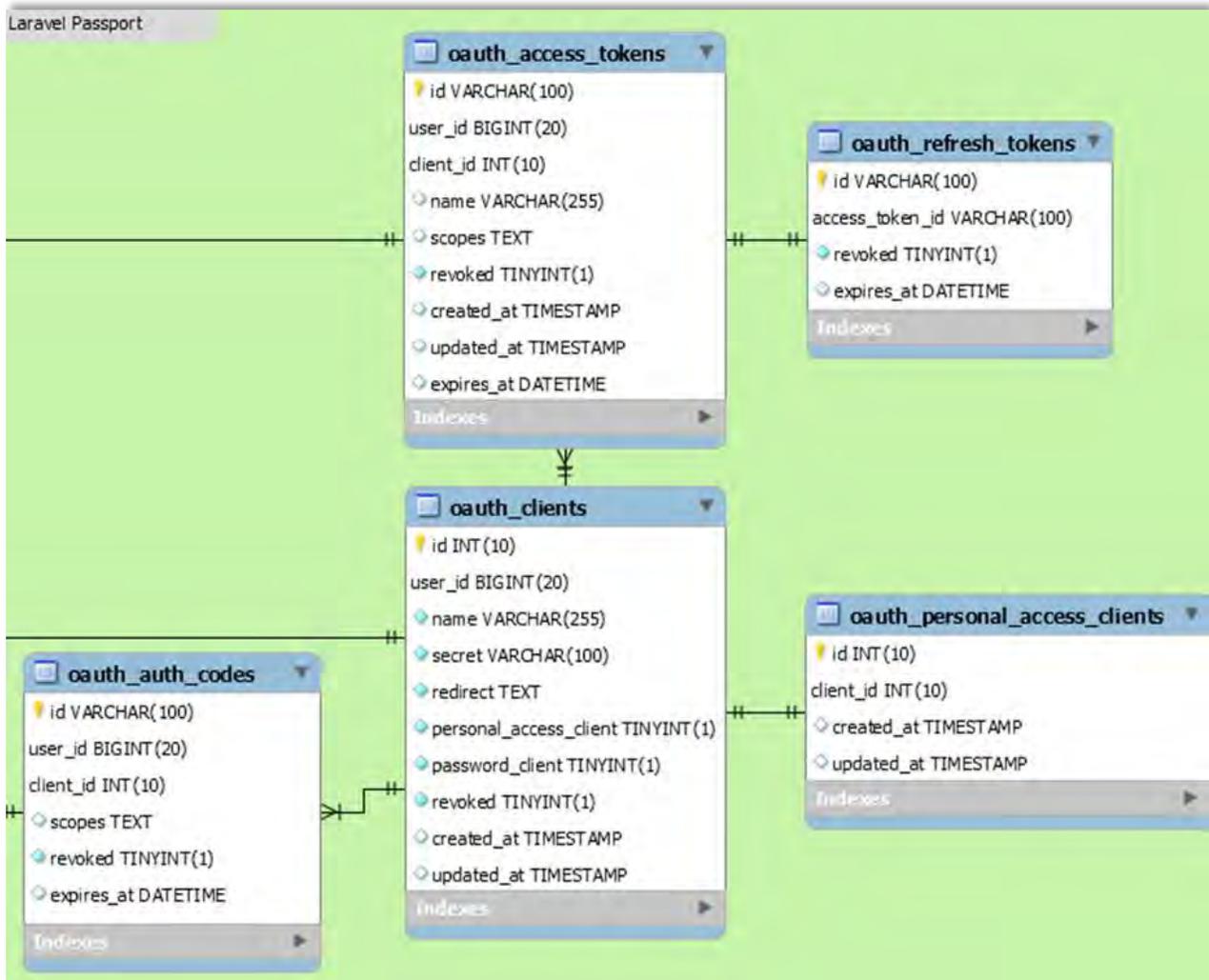


Figura 15. Diagrama relacional de Laravel Passport.

Spatie Permission es un paquete para Laravel encargado de controlar los privilegios o permisos de un usuario o grupo de usuarios a través de un sistema de roles y permisos, este paquete genera 5 tablas requeridas que almacenarán toda la información relacionada a este sistema de seguridad (ver Figura 16).

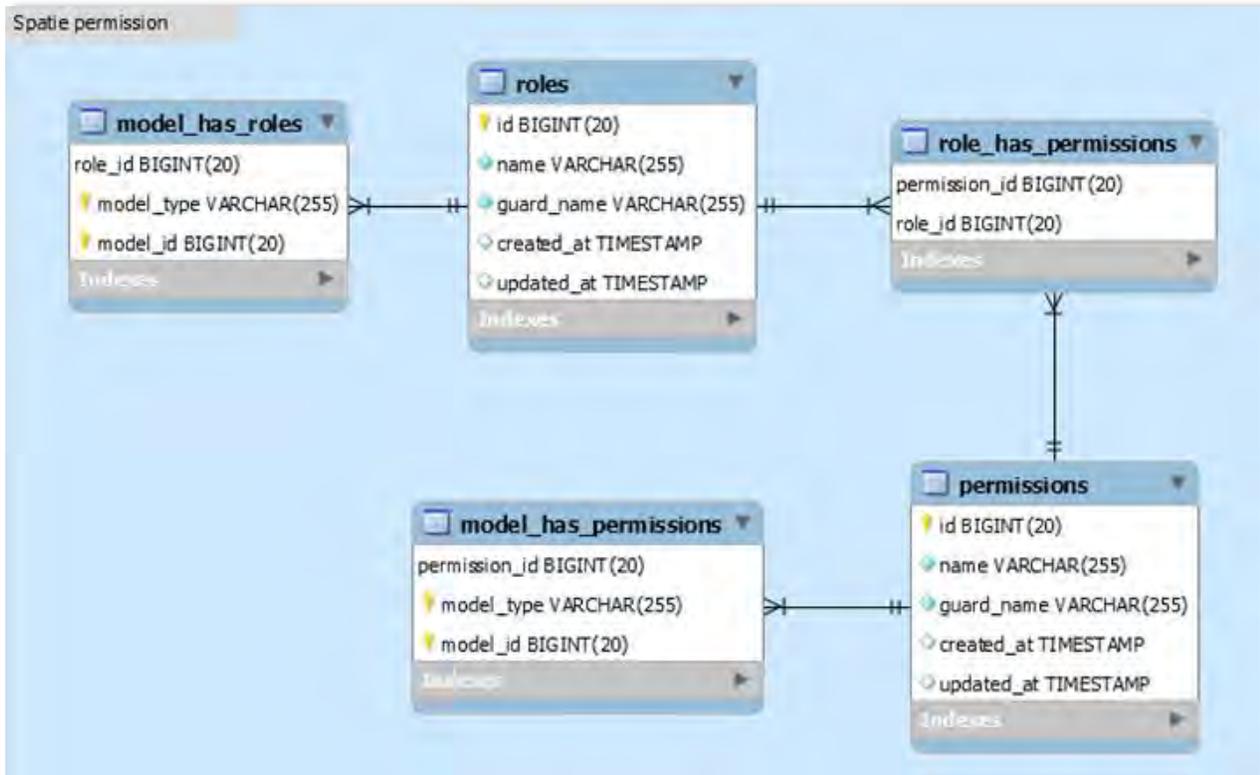


Figura 16. Diagrama relacional de Spatie Permission.

3.2.4 Diseño de las interfaces gráficas

Los bocetos iniciales de la aplicación móvil plasmaron los requisitos funcionales de los interesados (*stakeholders*) en interfaces de una aplicación para dispositivos móviles. En la Figura 17 se observan los primeros elementos funcionales que contiene la aplicación, una pantalla encargada de mostrar el **listado de unidades de salud**, que se muestra una vez que la pantalla de bienvenida sea removida, en la tercera captura de esta misma figura podemos observar el boceto de la pantalla que contiene el **mapa dinámico** de unidades de salud, solicitado por los stakeholders.

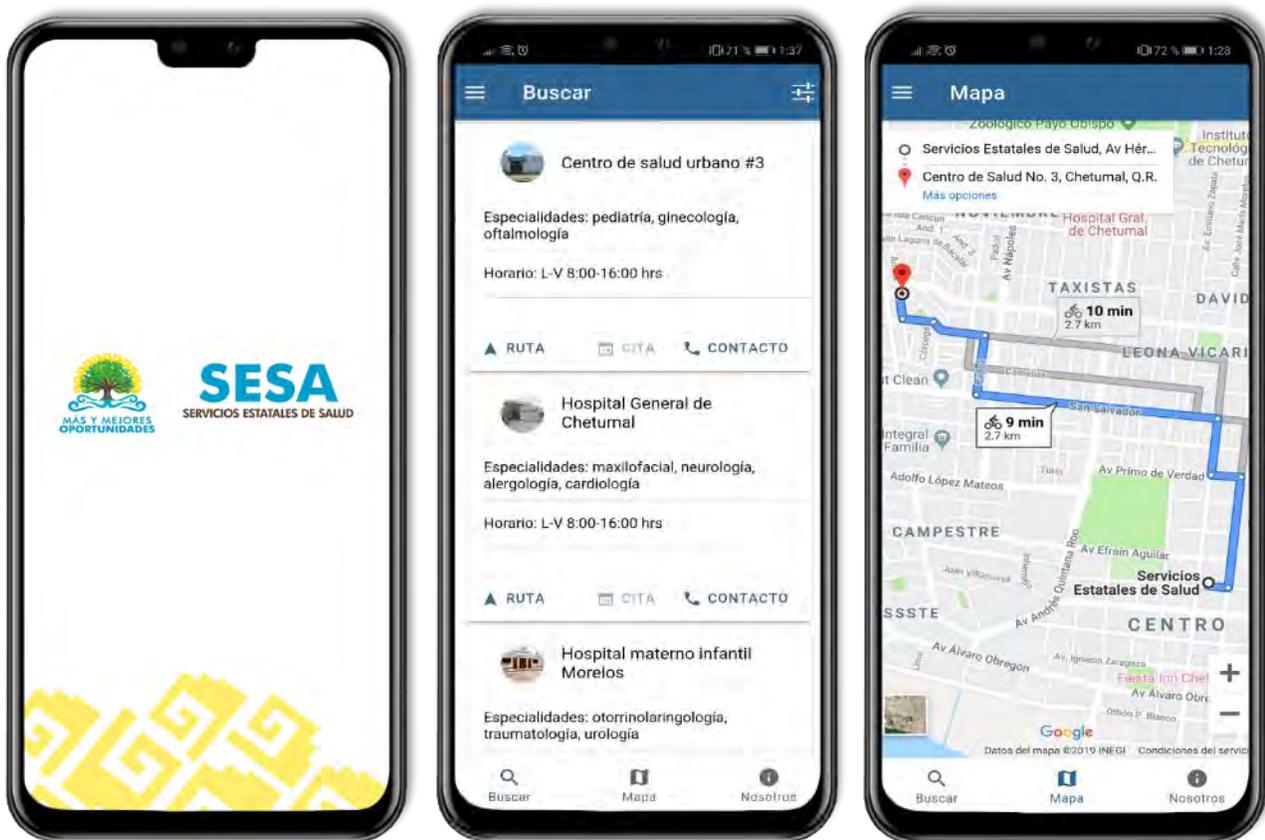


Figura 17. Pantalla de bienvenida, listado de unidades y mapa de la aplicación móvil.

Adicional a las pantallas anteriores, en la Figura 18 podemos observar tres pantallas requeridas que completan las funcionalidades principales de la aplicación móvil, la primera de ellas es la pantalla de **citas médicas** donde el usuario puede consultar la disponibilidad en la agenda médica de cada unidad de salud, con el fin de agendar una cita, la siguiente pantalla de la figura es la de **detalles**, encargada de mostrar información de interés de cada unidad de salud seleccionada, la tercera y última pantalla es la que permite al usuario **buscar** unidades de salud en específico.

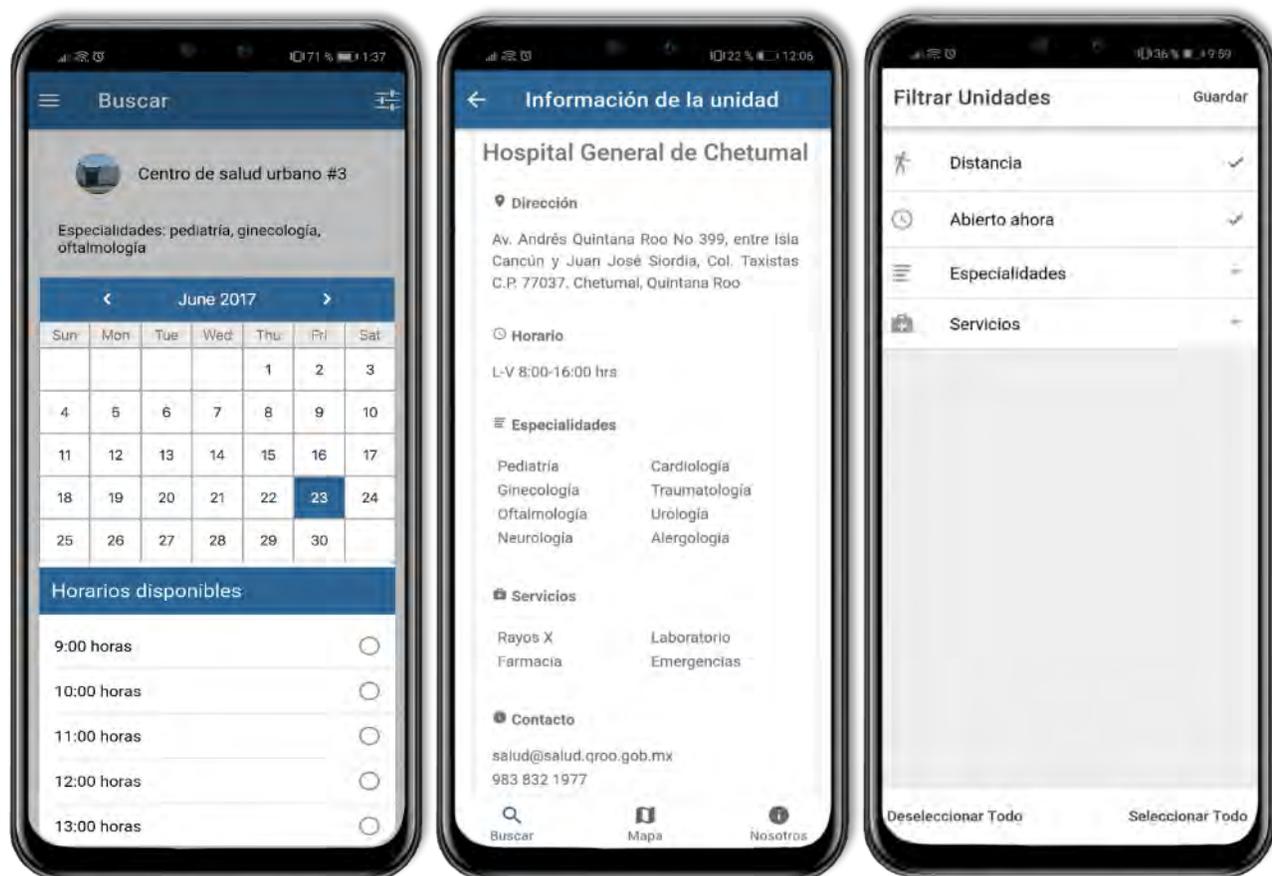


Figura 18. Agenda médica, detalles de la unidad de salud y filtros de búsqueda.

3.3 Fase de Codificación

3.3.1 Desarrollo de la pantalla de bienvenida

La pantalla de bienvenida tiene el objetivo de realizar acciones en segundo plano y preparar la información requerida por la aplicación móvil. Estas configuraciones se realizan en el archivo `app.component`, que es el primer controlador que se carga al iniciar la aplicación.

Para manejar seguridad en las peticiones web el servidor *back-end* de Laravel hace uso de *Laravel Passport*, un paquete de autenticación para APIs, cada consulta que se realice al back-end por la aplicación móvil deberá estar acompañada de un token de acceso válido, si la consulta se realiza sin un token o con un token inválido o expirado el servidor rechazará la petición web y devolverá un mensaje de error.

Cómo podemos ver en la Figura 19 la aplicación móvil manda sus credenciales de autenticación al servidor *back-end* a través de una petición web de tipo POST, si la respuesta es correcta el servidor regresará un texto en formato JSON (*JavaScript Object Notation*) con el token de acceso del usuario, una vez que tengamos el token se almacena en una variable global para seguir usándolo más adelante.

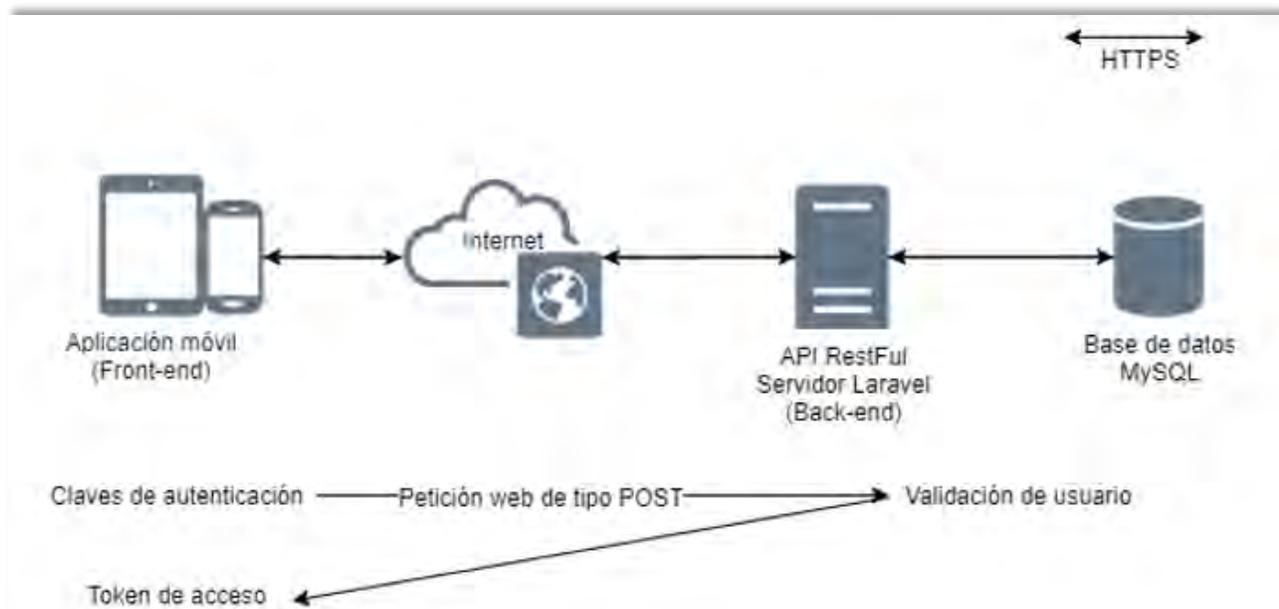


Figura 19. Proceso de autenticación con Laravel Passport.

Desde el archivo `app.component` se realiza la primera comunicación con el back-end para solicitar el token de acceso a través de una petición web, esta petición tiene un doble propósito, el primero de ellos es el que se mencionó anteriormente, autenticarse y recibir el token de acceso que se usará en todas las peticiones hacia el *back-end*; el segundo propósito es el de verificar la conectividad hacia el servidor, si el servidor no responde, no se iniciará la aplicación móvil y se mostrará un mensaje de error (ver Figura 20), este error puede aparecer por dos posibles problemas; Que el usuario no cuente con acceso a internet o que el servidor se encuentre fuera de servicio.

Geolocalización

Como se ha mencionado anteriormente, una parte fundamental de la aplicación móvil es el uso del posicionamiento geográfico para las funcionalidades dentro de la aplicación, para disponer de estos parámetros se utiliza el plugin de geolocalización que brinda Apache Cordova.

Este plugin proporciona información sobre la ubicación del dispositivo, como la latitud y la longitud. Las fuentes comunes de información de ubicación incluyen el Sistema de Posicionamiento Global (*GPS*) y la ubicación inferida de las señales de red, como la dirección IP, RFID, Wifi y direcciones MAC de Bluetooth, así como el cell *ID* de *GSM* / *CDMA*. (Ionic, 2020)



Figura 20. Error de conexión con el servidor.

```
TYPESCRIPT

import { Geolocation } from '@ionic-native/geolocation/ngx';

...

constructor(private geolocation: Geolocation) {}

...

this.geolocation.getCurrentPosition().then((resp) => {
  // resp.coords.latitude
  // resp.coords.longitude
}).catch((error) => {
  console.log('Error getting location', error);
});

let watch = this.geolocation.watchPosition();
watch.subscribe((data) => {
  // data can be a set of coordinates, or an error (if an error occurred).
  // data.coords.latitude
  // data.coords.longitude
});
```

Figura 21. Uso básico del plugin de geolocalización Apache Cordova. (Ionic, 2020)

Google Maps API

En este mismo controlador, mientras aún está la pantalla de bienvenida, se realiza la primera petición a la API de *Google Maps*. En la petición se solicita el Place ID de la localidad en la que se encuentra el usuario. El *Place ID* identifica de forma exclusiva un lugar en la base de datos de *Google Places* y en *Google Maps*.

En la base de datos de la aplicación móvil, se encuentran asociadas todas las localidades del Estado de Quintana Roo a los *Place IDs* de *Google Maps*. De esta forma, cuando el servicio de *Google Maps* nos regresa el Place ID del usuario se muestran las unidades de salud de su localidad al iniciar la aplicación móvil.

```
https://maps.googleapis.com/maps/api/geocode/json?latlng=48.714224,-73.961452  
&result_type=locality&key=YOUR_API_KEY
```

Figura 22. Reverse Geocoding filtrado por tipo. (Google, 2020)

El **Geocoding** es el proceso empleado para convertir direcciones en coordenadas geográficas (latitud, longitud), la aplicación utilizará el **Reverse Geocoding** convirtiendo coordenadas geográficas en direcciones.

En la Figura 22 se observa la estructura a seguir para realizar la petición web a la *API* de *Google Maps* solicitando el servicio de *Reverse Geocoding*, esta petición es acompañada de la latitud y longitud del usuario proporcionada por el plugin de geolocalización de Apache Cordova, para obtener exclusivamente el Place ID de la localidad se aplica un filtrado en la petición web por localidades y finalmente se añade a la petición el identificador de servicio de usuario (*API Key*).

Una vez recopilado el token de acceso, las coordenadas geográficas del usuario y el *Place ID* de su localidad, se cambia la pantalla de bienvenida (ver Figura 23) y aparece la pantalla de inicio de la aplicación móvil.



Figura 23. Resultado de la pantalla de bienvenida de la aplicación móvil.

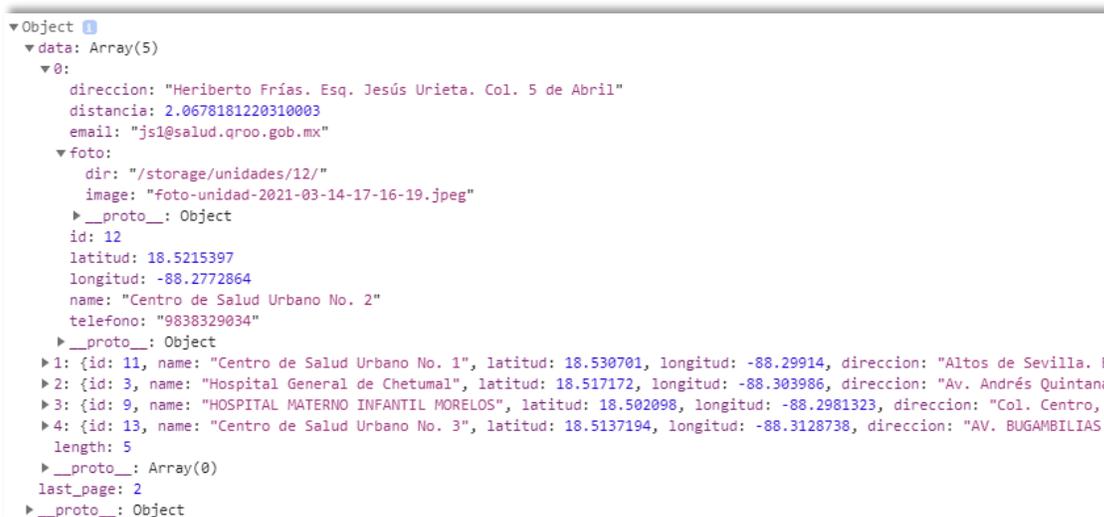
3.3.2 Desarrollo de la pantalla “Unidades”

La pantalla de unidades es la primera página que ve el usuario una vez que la aplicación móvil haya terminado de cargar contenido en segundo plano y se haya removido la pantalla de bienvenida.

Listado de unidades de salud

Este listado muestra las unidades de salud más cercanas al usuario y que, además, pertenecen a su localidad ², para obtener este listado se realizó una petición al back-end acompañada de la información que se obtuvo mientras estaba la pantalla de bienvenida (token de autenticación, *Place ID*, latitud y longitud).

El servidor responde a esta petición con un JSON, la respuesta obtenida incluye la información de cada una de las unidades de salud de la localidad del usuario y tiene la siguiente estructura:



```
▼ Object
  ▼ data: Array(5)
    ▼ 0:
      direccion: "Heriberto Frías. Esq. Jesús Urieta. Col. 5 de Abril"
      distancia: 2.0678181220310003
      email: "js1@salud.qroo.gob.mx"
      ▼ foto:
        dir: "/storage/unidades/12/"
        image: "foto-unidad-2021-03-14-17-16-19.jpeg"
        ▶ __proto__: Object
      id: 12
      latitud: 18.5215397
      longitud: -88.2772864
      name: "Centro de Salud Urbano No. 2"
      telefono: "9838329034"
      ▶ __proto__: Object
    ▶ 1: {id: 11, name: "Centro de Salud Urbano No. 1", latitud: 18.530701, longitud: -88.29914, direccion: "Altos de Sevilla. E..."
    ▶ 2: {id: 3, name: "Hospital General de Chetumal", latitud: 18.517172, longitud: -88.303986, direccion: "Av. Andrés Quintana..."
    ▶ 3: {id: 9, name: "HOSPITAL MATERNO INFANTIL MORELOS", latitud: 18.502098, longitud: -88.2981323, direccion: "Col. Centro, A..."
    ▶ 4: {id: 13, name: "Centro de Salud Urbano No. 3", latitud: 18.5137194, longitud: -88.3128738, direccion: "AV. BUGAMBILIAS..."
    length: 5
    ▶ __proto__: Array(0)
  last_page: 2
  ▶ __proto__: Object
```

Figura 24. Estructura de respuesta con el listado de unidades.

Esta colección de datos se debe recorrer con un ciclo, para mostrar una tarjeta con la información resumida de la unidad para cada elemento en el arreglo, además, esta respuesta es acompañada con un valor llamado “última página” esto debido a que se emplea una técnica de paginación.

² Disponible cuando la aplicación tiene acceso a la geolocalización del usuario.

Infinite Scroll

Este componente de *Ionic* invoca una acción cuando el usuario se desplaza una distancia especificada desde la parte inferior o superior de la página. *Infinite Scroll* hace posible la paginación en el listado de unidades. A medida que el usuario se desplace entre el listado se ejecutará un método que se encargará de realizar la petición al back-end por el listado de unidades de salud, solo que en este caso se añadirá a la petición el número de la página a solicitar.

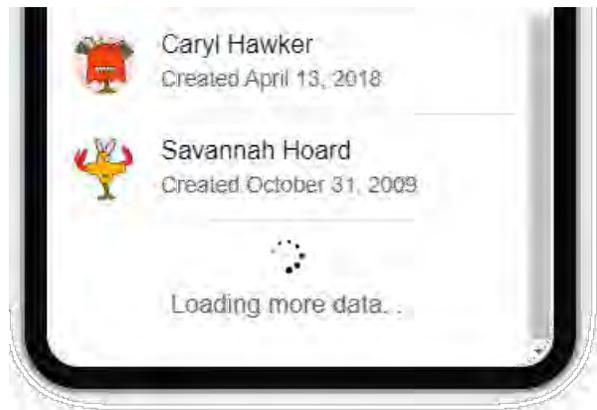


Figura 25. Componente *Infinite Scroll* de *Ionic*.

En la primera solicitud al entrar a la pantalla de inicio la respuesta con la información de unidades de salud solo incluye un número limitado de unidades correspondientes a la primera página, una vez que el usuario deslice hacia abajo y haya visto todas las unidades correspondientes a la primera página, el componente *Infinite Scroll* se encargará de ejecutar el método para solicitar las unidades correspondientes a la segunda página, y este proceso continuará hasta que se haya alcanzado el número máximo de páginas.

La paginación permite optimizar el uso de memoria del dispositivo móvil al evitar renderizar en el *DOM* todas las unidades de salud existentes en la localidad del usuario, la paginación nos sirve especialmente cuando el usuario esta filtrado unidades y la búsqueda de acuerdo con la selección del usuario pudiera incluir un gran número de unidades de salud.

De esta manera *Infinite Scroll* de *Ionic* carga en el *DOM* sólo los recursos que el usuario visualizará al deslizar sobre la pantalla principal y no consume memoria del dispositivo móvil de manera innecesaria al cargar el listado completo de unidades de salud que no todos los usuarios visualizarán.

Ion Refresher

Este componente de *Ionic* ofrece la funcionalidad de estirar (pull) para recargar el contenido de la página actual, se trata de una funcionalidad que tienen en común muchas aplicaciones móviles, este complemento ejecutará una acción cuando el usuario se encuentre al inicio de un listado y deslice este contenido haciendo un movimiento de arriba hacia abajo. En la Figura 26 podemos ver como se muestra esta funcionalidad de manera visual comúnmente para las plataformas Android y iOS, una vez que el usuario realice la acción anteriormente mencionada se mostrará el icono de recarga hasta que la aplicación termine de actualizar la información que se mostrará en pantalla.

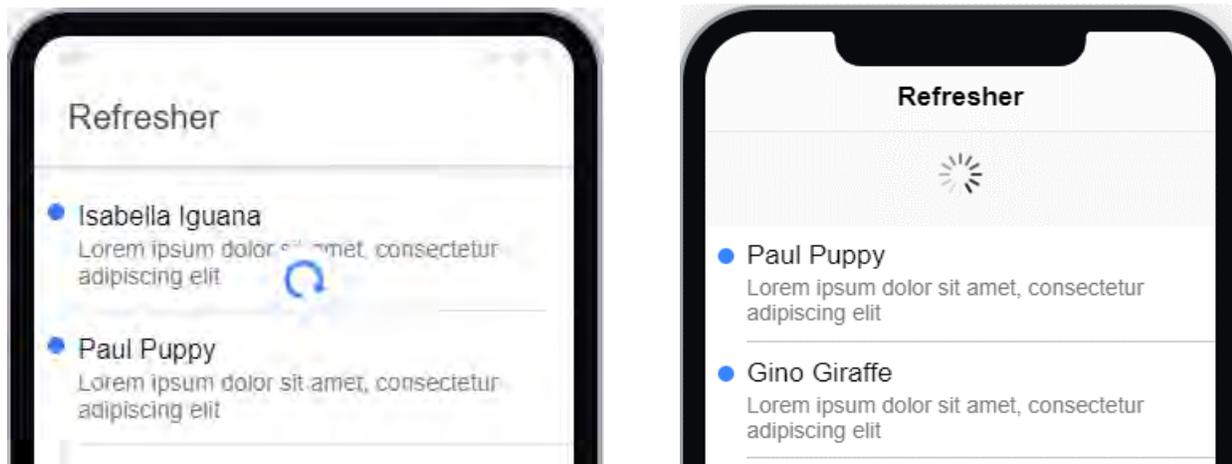


Figura 26. *Ion-Refresher* para Android y iOS. (*Ionic*, 2020)

El resultado visual final de hacer uso de este componente se puede ver en la Figura 27, en la primera pantalla se observa como aparece la flecha de actualizar a medida que el usuario realiza el movimiento de refrescar anteriormente mencionado.

Mientras se realizan las peticiones necesarias para actualizar el listado, aparece una pantalla de carga que se puede ver en la segunda captura de esta misma figura. Los esqueletos de carga le permiten al usuario saber que el contenido fue solicitado con éxito y está siendo procesado para ser renderizado. Las animaciones de carga se logran gracias al componente “*ion-skeleton-text*”.

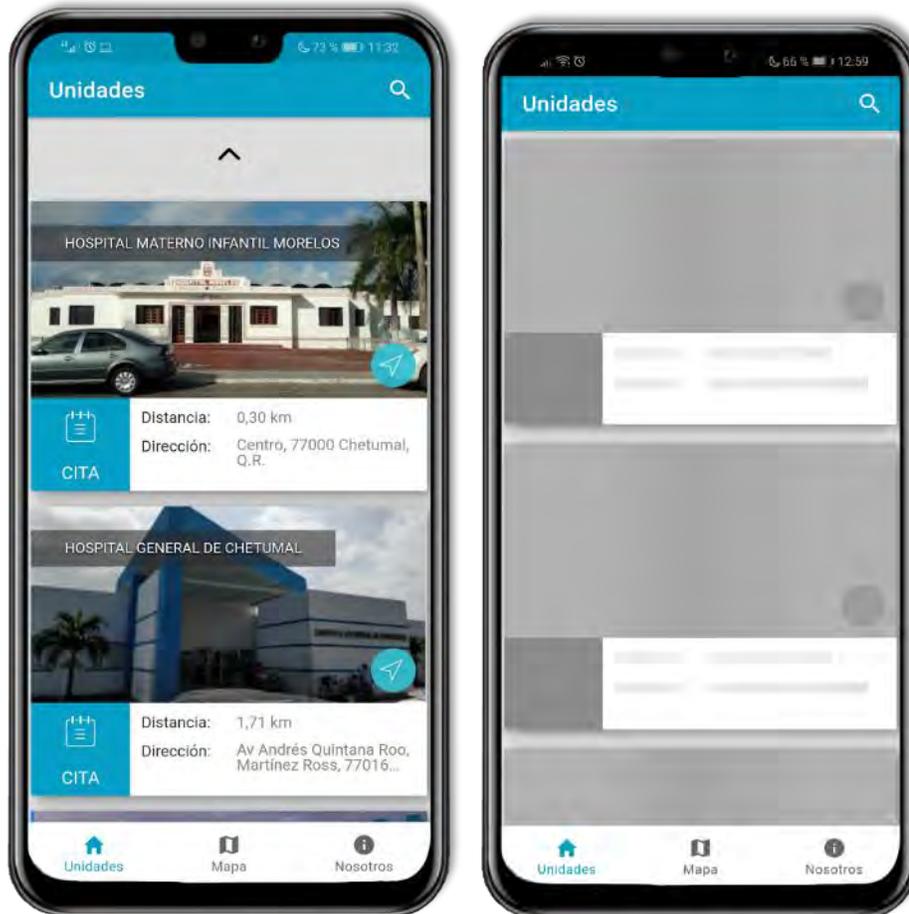


Figura 27. Resultado del componente Ion Refresher en la aplicación móvil.

Cuando se ejecuta el método creado por esta función se realiza nuevamente la petición hacia el *back-end* de las unidades de salud, pero se actualiza la ubicación del usuario, por lo tanto, la única información que cambiará al momento de actualizar el listado será la relacionada a la distancia actual entre el usuario y las unidades.

De igual manera, cuando exista algún filtro aplicado, se refrescará la petición con todos los filtros aplicados, pero nuevamente la única información que cambiará será la distancia entre las unidades de salud y el usuario. Sin embargo, si llegará a existir algún cambio en la base de datos que contiene la información de cada unidad de salud, esta información también cambiaría al terminar de actualizarse la pantalla.

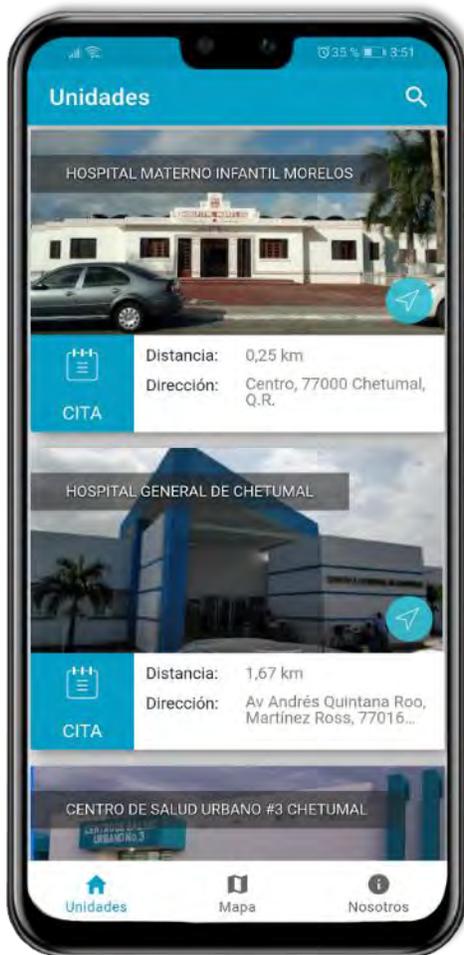


Figura 28. Pantalla unidades de la aplicación móvil.

El resultado final de esta pantalla que incluye el listado de unidades de salud se puede ver en la Figura 28, este listado contiene una tarjeta por cada unidad de salud, que incluye una imagen de la unidad, el nombre completo, dirección y distancia, además se cuenta con un acceso rápido para obtener la ruta a la unidad de salud y para agendar una cita médica.

Al hacer clic sobre cada una de estas tarjetas se realizará el cambio a una nueva pantalla que muestra todos los detalles de esa unidad.

Ruta hacia la unidad de salud

Para la funcionalidad de trazar la ruta hacia la unidad de salud se hace uso del plugin “*Launch Navigator*”, este plugin utiliza la aplicación nativa de mapas del dispositivo y traza una ruta hacia un destino.

```
launchnavigator.isAppAvailable(launchnavigator.APP.GOOGLE_MAPS, function(isAvailable){
  var app;
  if(isAvailable){
    app = launchnavigator.APP.GOOGLE_MAPS;
  }else{
    console.warn("Google Maps not available - falling back to user selection");
    app = launchnavigator.APP.USER_SELECT;
  }
  launchnavigator.navigate("London, UK", {
    app: app
  });
});
```

Figura 29. Ejemplo de uso del plugin “Launch Navigator”.

En el fragmento de código anterior (ver Figura 29) se ve un ejemplo de uso del plugin “*Launch Navigator*”, para este ejemplo, se verifica que el dispositivo móvil cuente con la aplicación de mapas de *Google Maps*, en caso de que el dispositivo no cuente con esta aplicación, se le dará al usuario la opción de elegir la aplicación de mapas que desea usar y que actualmente este instalada en su dispositivo móvil.

Una vez definida se hace el llamado para ejecutar la aplicación de mapas, este llamado va acompañado con el lugar de destino, en el ejemplo de la Figura 29 el lugar de destino está marcado en “London, UK”, para el caso específico de la aplicación móvil, se reemplazará este destino por las coordenadas geográficas de la unidad de salud en la que el usuario haya seleccionado la opción de “ruta”.

En la Figura 30 podemos ver el resultado final donde el plugin lanzó de manera exitosa la aplicación de mapas de *Google Maps* con una ruta que tiene como destino el “Hospital Materno Infantil Morelos”.

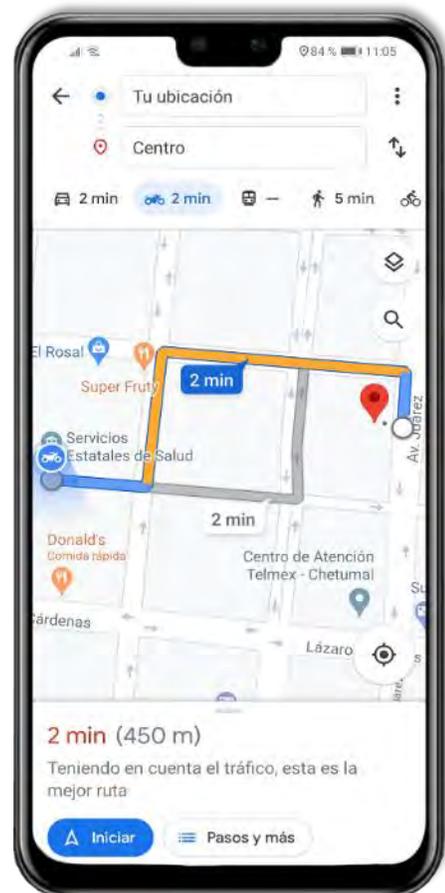


Figura 30. Ejemplo de ruta hacia una unidad de salud con Google Maps.

3.3.3 Desarrollo de la funcionalidad de filtros

Los filtros le permiten al usuario hacer una búsqueda ágil para encontrar una unidad de salud, en la Figura 31 se puede ver el resultado final de la pantalla para la selección de filtros de búsqueda, el usuario tiene la posibilidad filtrar por localidades del estado de Quintana Roo, tipo de unidad, especialidades y servicios.

El usuario tiene la opción de seleccionar una o más opciones de cada una de las categorías de filtros, incluso puede combinar opciones de filtros diferentes. La única condición para poder realizar la búsqueda es que el usuario haya seleccionado al menos uno de los filtros existentes, en caso contrario la aplicación móvil mostrará un aviso notificando la ausencia en la selección de filtros al momento de realizar la búsqueda.

Cuando el usuario selecciona los filtros de búsqueda, estos se almacenan en una variable con formato JSON tal y como se observa en la Figura 32.

```
var filtros = {  
  'localidades': ['id_localidad_ejemplo'],  
  'tipos_unidades': ['id_tipo_unidad_ejemplo'],  
  'especialidades': ['id_especialidad_ejemplo'],  
  'servicios': ['id_servicio_ejemplo'],  
  'latitud': '18.538823',  
  'longitud': '-88.270657'  
}
```

Figura 32. Variable que contiene los filtros seleccionados del usuario.

Si se cuenta con acceso a la ubicación del dispositivo, esta información también se añade a la variable filtros que será enviada al *back-end* a través de una petición web. Una vez que el *back-end* regrese una respuesta exitosa con la información de las unidades de salud se redireccionará nuevamente al usuario a la pantalla de unidades.

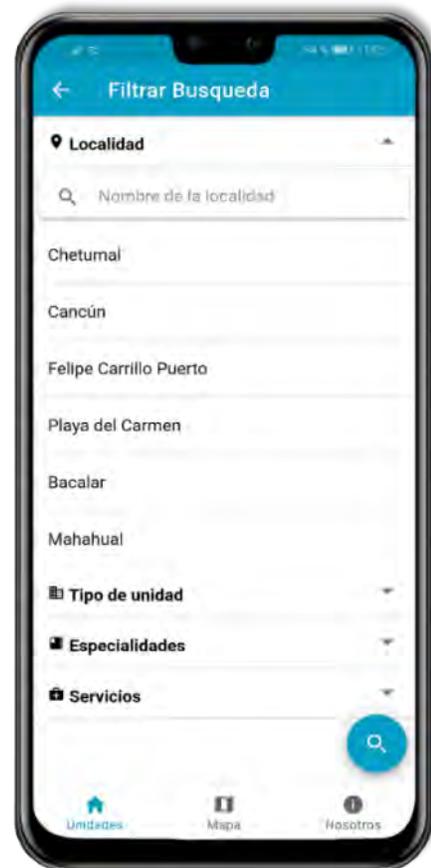


Figura 31. Pantalla de búsqueda de unidades de salud.

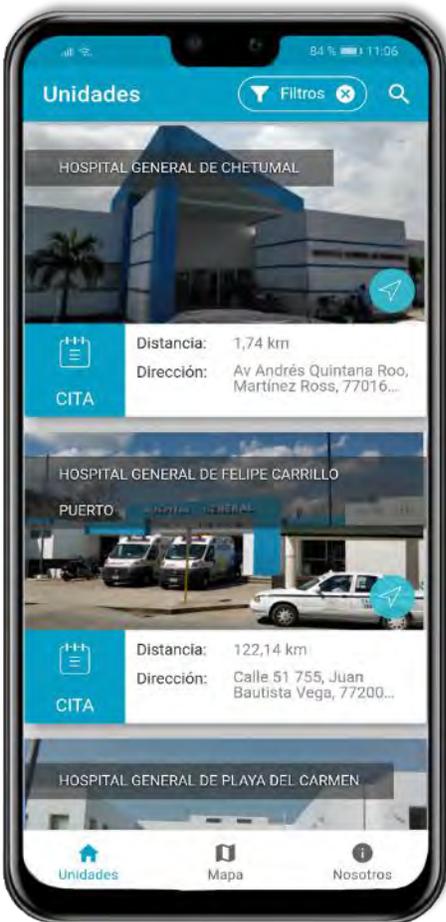


Figura 33. Filtrado de unidades por tipo de unidad.

3.3.4 Desarrollo de la pantalla “Detalles”

La pantalla de “Detalles” fue desarrollada con el objetivo de mostrar toda la información que podría ser de utilidad para el usuario, referente a la unidad de salud seleccionada. Para acceder a esta pantalla se debe tocar sobre la tarjeta de alguna unidad de salud desde la pantalla “Unidades” o “Mapas”.

Para obtener la información de la unidad en cuestión se hace una petición web de tipo *GET* al *back-end*, petición que va acompañada del identificador único de unidad, esta solicitud se realiza al iniciar la carga de la pantalla de detalles, si la respuesta es exitosa, el servidor regresa un archivo *JSON* que contiene la información de la unidad de salud y sigue la estructura que se observa en la Figura 34.

En la pantalla de unidades será mostrado el listado de unidades que cumpla con todos los filtros que haya aplicado el usuario. En la Figura 33 las unidades de salud se encuentran filtradas por tipo de unidad, para este ejemplo en específico se seleccionó el tipo de unidad “Hospital general”.

Cuando existe un filtro activo aparece un nuevo botón en la pantalla de unidades, este botón nos indica que actualmente hay un filtro activo y al tocarlo eliminará el filtro de búsqueda y regresará el listado a su punto inicial.

```
atencion_inicio: "07:00:00"
atencion_termino: "03:00:00"
cat_nivel_id: 3
clave_clues: "QRSSA001843"
codigo_postal: 77040
direccion: "Km 6, carretera Federal Chetumal - Bacalar S/N, Colonia Santa Isabel CP 77040, Chetumal, Quintana Roo."
distancia: null
email: "direccion.hech@gmail.com"
especialidades: [{-}]
fecha_construccion: "2019-02-16"
fecha_inicio_operaciones: "2019-02-16"
fotos: (4) [{-}, {-}, {-}, {-}]
id: 4
latitud: 18.5077
longitud: -88.347433
name: "HOSPITAL DE ESPECIALIDADES DE CHETUMAL"
servicios: Array(1)
  ▼ 0:
    descripcion: "Terapia de radiación (también llamada radioterapia) es un tratamiento del cáncer que usa altas dos
  ▼ horarios: Array(1)
    ▼ 0:
      doctores: ["Marlene Olvera Dávalos"]
      domingo: 0
      hora_inicio: "07:00:00"
      hora_termino: "15:00:00"
      jueves: 1
      lunes: 1
      martes: 1
      miercoles: 1
      sabado: 0
      viernes: 1
      ▶ __proto__: Object
      length: 1
      ▶ __proto__: Array(0)
      id: 19
      img: "radioterapia_1615162740.jpeg"
      name: "Radioterapia"
      ▶ __proto__: Object
      length: 1
      ▶ __proto__: Array(0)
      telefono: "3331598322"
```

Figura 34. Respuesta de la solicitud con la información detallada de la unidad.

El resultado final de la pantalla de detalles se puede ver en la Figura 35 donde se plasma toda la información obtenida de la petición web organizada amigablemente para el usuario. La primera sección de esta pantalla corresponde al encabezado, donde se encuentra un botón de acciones con las opciones de; Llamar a la unidad de salud, enviar un correo electrónico, agendar una cita y trazar la ruta hacia la unidad.

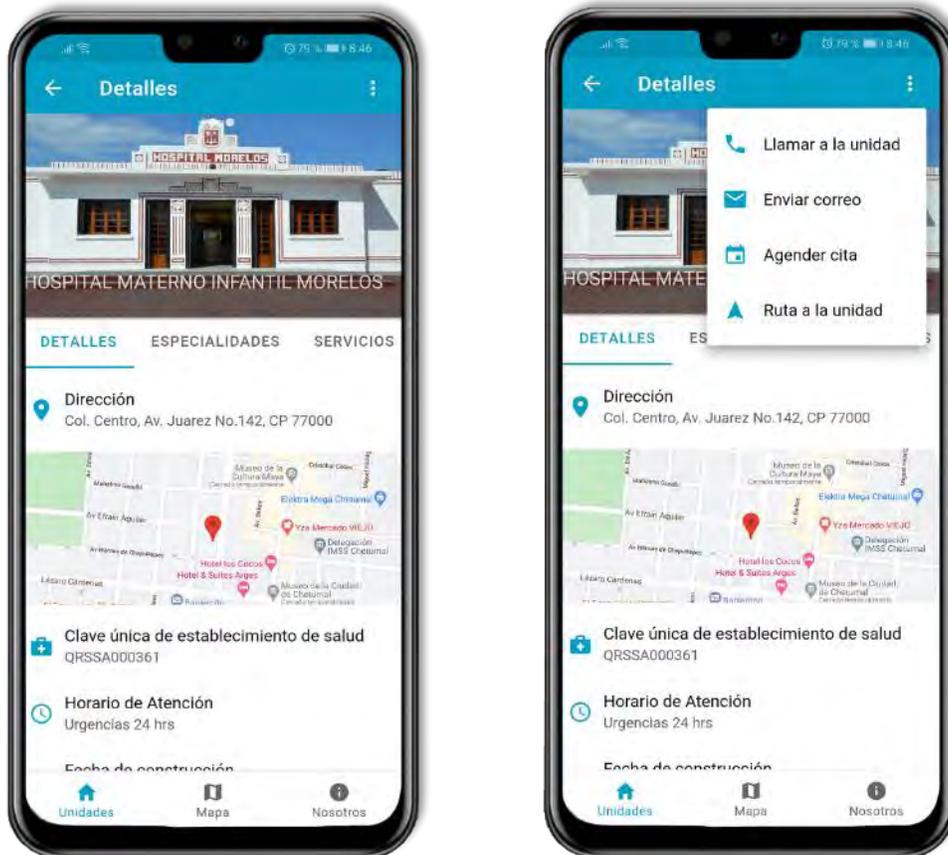


Figura 35. Pantalla de detalles de la aplicación móvil y menú de acciones.

La siguiente sección de esta tarjeta, se trata de una colección de fotos de la unidad de salud, que tiene como objetivo orientar al ciudadano al llegar a la dirección en la que se encuentra su unidad destino. Al tocar cualquiera de estas fotos se visualizarán en pantalla completa.

Finalmente llegamos a la parte más importante de esta pantalla, donde se le mostrará al usuario la información detallada de la unidad y que se presenta en tres secciones principales. En la primera sección es mostrada la dirección de la unidad de salud, seguido de un mapa estático de *Google Maps* con una marca sobre la unidad.

La API de *Static Maps* permite incrustar una imagen de *Google Maps* en una página web sin necesidad de JavaScript ni ninguna carga de página dinámica. El servicio *Maps Static API* crea un mapa en función de los parámetros de URL enviados a través de una solicitud HTTP estándar y devuelve el mapa como una imagen. (Google, 2020)

En esta misma sección se incluye la clave única de establecimiento de salud (CLUES), el horario de atención que brinda la unidad, fecha de construcción y fecha de inicio de operaciones. Estos datos localizados en la primera sección pueden aumentar o ser reemplazados por algún otro dato de relevancia de la unidad de salud que consideren conveniente mostrar los interesados de la aplicación móvil.



Figura 36. Sección de especialidades en la pantalla de detalles.

La segunda sección de esta pantalla está dedicada a las especialidades médicas, como se visualiza en la Figura 36, se trata de un apartado que contiene pequeños recuadros con una imagen de referencia por cada especialidad acompañado de su nombre.

De esta forma el usuario puede identificar fácilmente cada especialidad con la que cuenta la unidad de salud. Al tocar cualquier recuadro se abrirá una ventana emergente con más información de la especialidad médica.

Las ventanas emergentes son un componente de *Ionic* que llevan el nombre de “*ion-modal*”. La documentación oficial de *Ionic* describe este componente como “un cuadro de diálogo que aparece sobre el contenido de la aplicación y la aplicación debe descartarlo antes de que se pueda reanudar la interacción. Es útil como un componente de selección cuando hay muchas opciones para elegir, o cuando se filtran elementos en una lista, así como muchos otros casos de uso.”



Figura 37. Modal para especialidades médicas.

En la Figura 38 se visualiza un modal adicional que es mostrado al momento de tocar sobre un horario de atención, este nuevo modal mostrará un listado con el nombre del personal médico en turno que imparte la especialidad en el horario seleccionado, si la información del personal médico no está disponible solo se verá el mensaje de "Personal Médico En Turno."

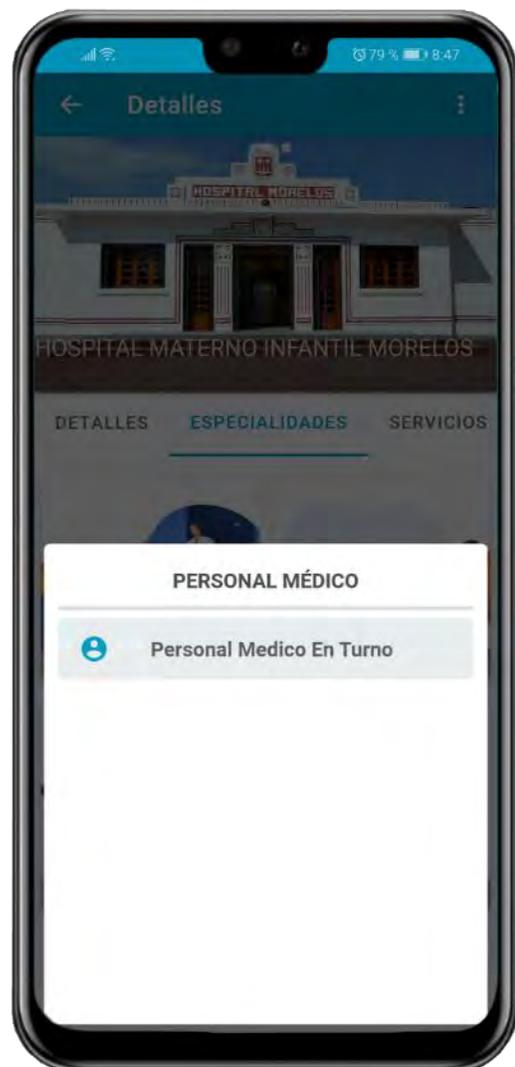


Figura 38. Modal de personal médico que imparte una especialidad o servicio.

La tercera y última sección de esta pantalla (ver Figura 39) está dedicada a los servicios médicos, de igual manera que la sección anterior la información es presentada al usuario a través de pequeños recuadros con una imagen referente al servicio y su nombre.

El modal para los servicios se puede ver en la Figura 40 y presenta la información de la misma manera que es mostrada en la sección de especialidades médicas, se muestra un listado de horarios en los que se presta el servicio y una descripción de este.



Figura 39. Sección de servicios en la pantalla de detalles.



Figura 40. Modal para servicios médicos.

De igual manera se incluyen botones para llamar o enviar un correo electrónico al área del servicio médico. Al tocar sobre la opción de llamar se abrirá la aplicación de llamadas con el número del área pre marcado, de manera similar al tocar sobre la opción de correo, se abrirá la aplicación de correo electrónico con un nuevo correo con destinatario al área donde se imparte el servicio médico.

3.3.5 Desarrollo de la pantalla “Mapa”

En este apartado se aborda la integración de la *API* de *Google Maps* para crear un mapa dinámico de las unidades de salud dentro de la aplicación móvil.

Para hacer uso de este servicio, Google pone a disposición *Google Maps Platform* que constituye un conjunto de *API* y SDK que se administran desde la consola de Google Cloud Platform, a través de esta plataforma el usuario final puede llevar un control total de todos los servicios de geolocalización ofertados por *Google Maps*.

Después del registro y la creación de una cuenta de facturación se podrá hacer uso del conjunto de *API* y SDK. La facturación se realiza de manera mensual y el cargo dependerá del volumen de consultas que se hayan realizado, aplicándose la tarifa actual en cada una de ellas.

SKU	\$200 de crédito mensual Uso gratuito equivalente	Rango de volumen mensual (Precio por miles)		
		De 0 a 100,000	De 100,001 a 500,000	Más de 500,001
Mobile Native Static Maps	Cargas ilimitadas	\$0.00	\$0.00	COMUNÍCATE CON VENTAS
Mobile Native Dynamic Maps	Cargas ilimitadas	\$0.00	\$0.00	para obtener descuentos por volumen.
Embed	Cargas ilimitadas	\$0.00	\$0.00	
Embed Advanced	Hasta 14,000 cargas	\$14.00	\$11.20	
Static Maps	Hasta 100,000 cargas	\$2.00	\$1.60	
Dynamic Maps	Hasta 28,000 cargas	\$7.00	\$5.60	
Static Street View	Hasta 28,000 panorámicas	\$7.00	\$5.60	
Dynamic Street View	Hasta 14,000 panorámicas	\$14.00	\$11.20	

Figura 41. Tarifas para todas las APIs de Mapas. (Google, 2020)

SKU	\$200 de crédito mensual Uso gratuito equivalente	Rango de volumen mensual (Precio por miles)		
		De 0 a 100,000	De 100,001 a 500,000	Más de 500,001
Directions	Hasta 40,000 llamadas	\$5.00	\$4.00	COMUNÍCATE CON VENTAS para obtener descuentos por volumen.
Directions Advanced	Hasta 20,000 llamadas	\$10.00	\$8.00	
Distance Matrix	Hasta 40,000 elementos	\$5.00	\$4.00	
Distance Matrix Advanced	Hasta 20,000 elementos	\$10.00	\$8.00	
Roads - Route Traveled	Hasta 20,000 llamadas	\$10.00	\$8.00	
Roads - Nearest Road	Hasta 20,000 llamadas	\$10.00	\$8.00	
Roads - Speed Limits	Hasta 2,000 elementos	\$20.00	\$16.00	

Figura 42. Tarifas para todas las APIs de Rutas. (Google, 2020)

Find Place + datos básicos + datos de contacto + datos atmosféricos	Hasta 8,000 llamadas	\$17.00 \$0.00 \$3.00 \$5.00	\$13.60 \$0.00 \$2.40 \$4.00	
Costo total:		\$25.00	\$20.00	
Find Place – ID only	Ilimitado	\$0.00	\$0.00	
Llamadas a Find Current Place. Costo basado en la llamada + los datos				
Find Current Place + datos básicos	Hasta 6,500 llamadas	\$30.00 \$0.00	\$24.00 \$0.00	
Costo total:		\$30.00	\$24.00	
Find Current Place + datos básicos + datos de contacto + datos atmosféricos	Hasta 5,000 llamadas	\$30.00 \$0.00 \$3.00 \$5.00	\$24.00 \$0.00 \$2.40 \$4.00	
Costo total:		\$38.00	\$30.40	
Otras solicitudes a Places (Nota: Las solicitudes a Nearby Search y Text Search muestran, de forma predeterminada, datos de todos los tipos, lo cual activa todos los SKU de datos).				
Places Photo	Hasta 28,000 llamadas	\$7.00	\$5.60	COMUNICATE CON VENTAS
Places - Nearby Search + datos básicos + datos de contacto + datos atmosféricos	Hasta 5,000 llamadas	\$32.00 \$0.00 \$3.00 \$5.00	\$25.60 \$0.00 \$2.40 \$4.00	para obtener descuentos por volumen.
Costo total:		\$40.00	\$32.00	
Places - Text Search + datos básicos + datos de contacto + datos atmosféricos	Hasta 5,000 llamadas	\$32.00 \$0.00 \$3.00 \$5.00	\$25.60 \$0.00 \$2.40 \$4.00	
Costo total:		\$40.00	\$32.00	
Otros tipos de solicitudes relacionadas con Places				
Geocoding	Hasta 40,000 llamadas	\$5.00	\$4.00	COMUNICATE CON VENTAS
Geolocation	Hasta 40,000 llamadas	\$5.00	\$4.00	para obtener descuentos por volumen.
Time Zone	Hasta 40,000 llamadas	\$5.00	\$4.00	
Elevation	Hasta 40,000 llamadas	\$5.00	\$4.00	

Figura 43. Fracción de tarifas para las API de Places.

Anteriormente se mencionó la función de *Reverse Geocoding* y *Static Maps* de la API de *Google Maps* en la pantalla de bienvenida y la pantalla de detalles respectivamente, la tercera y última función que se emplea de la API de *Google Maps* en el desarrollo de este proyecto es *Dynamics Maps*, usada en esta pantalla.

De acuerdo con la guía de desarrollador de Google, 2020. La API JavaScript de Maps para *Dynamics Maps* permite al desarrollador personalizar mapas con su propio contenido e imágenes para mostrar en páginas web y dispositivos móviles. Esta API presenta cuatro tipos de mapas básicos (mapa vial, satélite, híbrido y terreno) que pueden ser modificados utilizando capas, estilos, controles, eventos, servicios y bibliotecas.

Propiedades del mapa

Antes de que el mapa sea mostrado en pantalla, son configurados todos los elementos de interfaz de usuario que contendrá el mapa, estos elementos permiten la interacción del usuario con el mapa y son conocidos como “controles”.

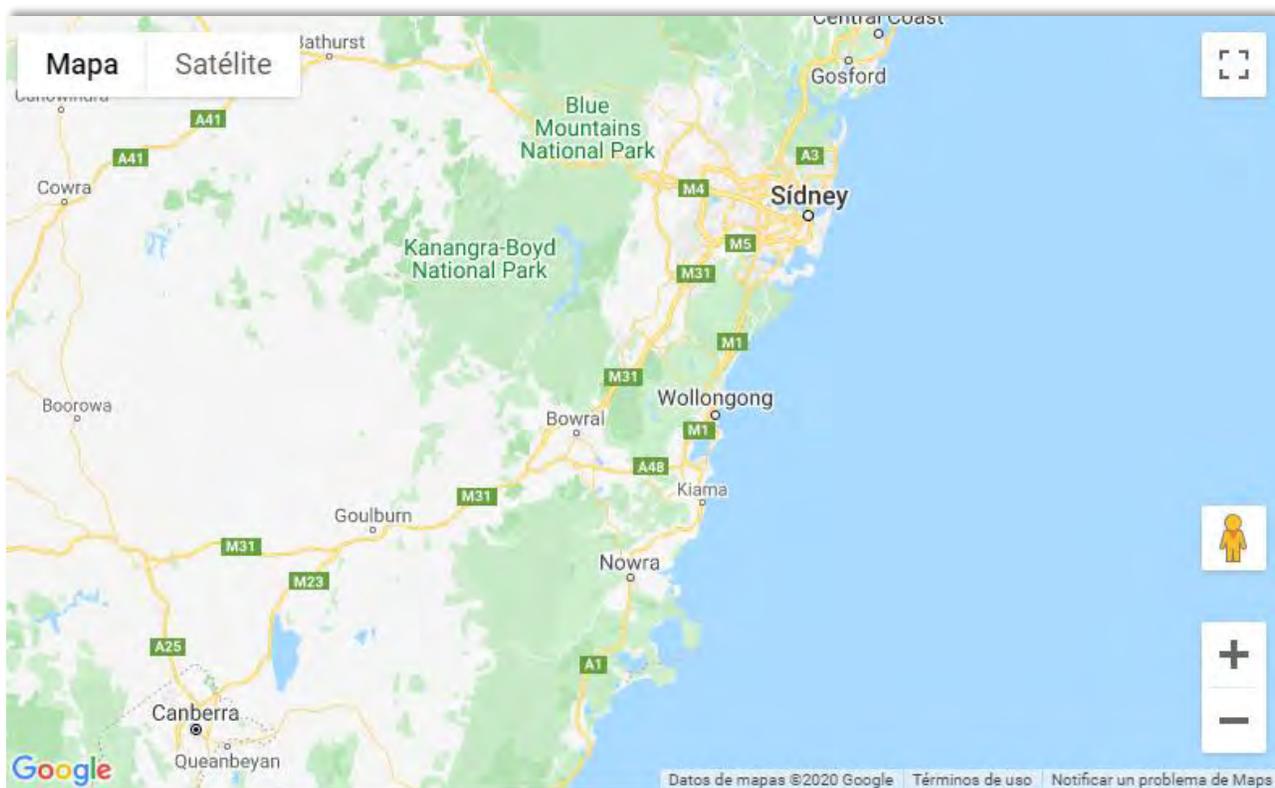


Figura 44. Conjunto predeterminado de controles para un mapa nuevo.

Al crear un nuevo mapa se establece una configuración predeterminada de controles, como se ve en la Figura 44. El listado completo de controles que se pueden incluir en un mapa es mostrado a continuación:

- El **control de zoom** muestra los botones "+" y "-" para cambiar el nivel de zoom del mapa. Este control aparece de forma predeterminada en la esquina inferior derecha del mapa.
- El **control de tipo de mapa** está disponible a través de un menú desplegable que permite al usuario elegir un tipo de mapa (mapa vial, satélite, híbrido y terreno). Este control aparece de forma predeterminada en la esquina superior izquierda del mapa.
- El **control de Street View** contiene un icono que se puede arrastrar al mapa para habilitar Street View. Este control aparece de forma predeterminada cerca de la esquina inferior derecha del mapa.
- El **control Rotar** proporciona una combinación de opciones de inclinación y rotación para mapas que contienen imágenes oblicuas. Este control aparece de forma predeterminada cerca de la esquina inferior derecha del mapa.
- El **control Escala** muestra un elemento de escala del mapa. Este control está deshabilitado por defecto.
- El **control de pantalla completa** ofrece la opción de abrir el mapa en modo de pantalla completa. Este control está habilitado de forma predeterminada en dispositivos de escritorio y móviles.³

³ Nota: iOS no es compatible con la función de pantalla completa. Por lo tanto, el control de pantalla completa no es visible en dispositivos iOS.

La configuración de controles se realiza al momento de crear un nuevo mapa, para este proyecto son removidos todos los controles a excepción del control de zoom, que es reubicado a la esquina superior derecha.

```
return new google.maps.Map(document.getElementById('map'), {
  center: center,
  zoom: 13,
  zoomControl: true,
  zoomControlOptions: {
    position: google.maps.ControlPosition.TOP_LEFT
  },
  mapTypeControl: false,
  scaleControl: false,
  streetViewControl: false,
  rotateControl: false,
  fullscreenControl: false
});
```

Figura 45. Configuración de controles para el mapa de la pantalla "Mapa".

En la Figura 45 además de la configuración relacionada con los controles del mapa y el nivel del zoom se configura las coordenadas en las que se encontrará centrado el mapa. Estas coordenadas son las de la primera unidad que se encuentre en el listado de unidades de la pantalla "unidades", esto debido a que la pantalla de mapa está estrechamente ligada a la pantalla unidades.

Tarjetas de información

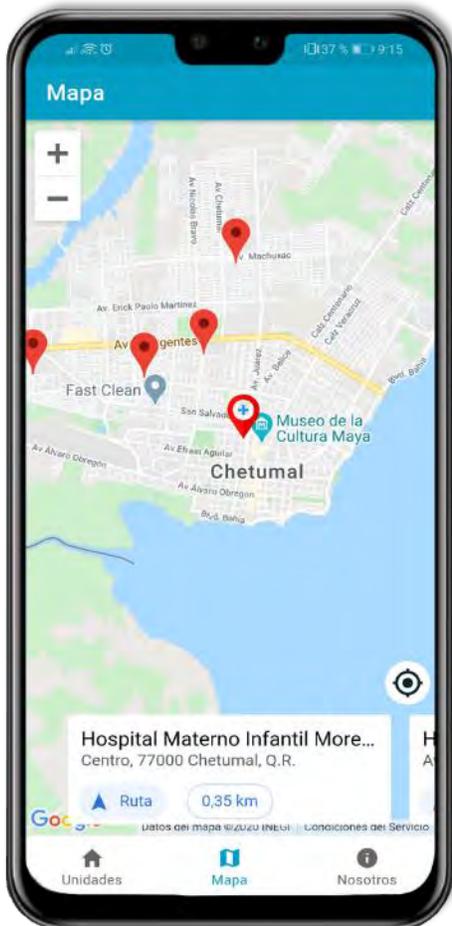


Figura 46. Pantalla de "Mapa", vista de tarjetas de información minimizada.

Las tarjetas de información son un elemento importante en la pantalla de mapa, se encargan de mostrar un resumen de la información de la unidad de salud seleccionada.

Cada marca en el mapa representa una unidad de salud, esta marca a su vez está vinculada a una tarjeta de información, de tal manera que al deslizar las tarjetas de información el marcador activo cambiará al que esté vinculado con la tarjeta actual, de manera viceversa, al tocar sobre cualquier marcador, se volverá visible la tarjeta vinculada a ese marcador.

Los marcadores y las tarjetas de información se pueden observar en la Figura 46, captura del diseño final de la pantalla "Mapa".

Marcadores

Los marcadores tienen el propósito de identificar una ubicación en el mapa y están diseñados para ser interactivos. De manera predeterminada, reciben los eventos de “clic”, esto permite crear un detector de eventos para llamar a una acción al tocar cualquier marcador.

La aplicación realiza tres acciones en específico cuando se pulsa sobre un marcador; La primera de ellas se mencionó anteriormente y consiste en mostrar en pantalla la tarjeta vinculada al marcador que se haya tocado. La segunda acción consiste en realizar un zoom y centrar el mapa en la posición del marcador pulsado, finalmente, la última acción consiste en ampliar la tarjeta de información que hará visible la imagen de la unidad de salud, siempre y cuando aún no se encuentren maximizadas las tarjetas (ver Figura 47), de manera predetermina al entrar al mapa por primera vez estas imágenes no son mostradas (ver Figura 46) con el propósito de no reducir la visión del usuario con el mapa, pero se harán visible una vez que se pulse cualquier marcador o al deslizar la tarjeta hacia arriba.

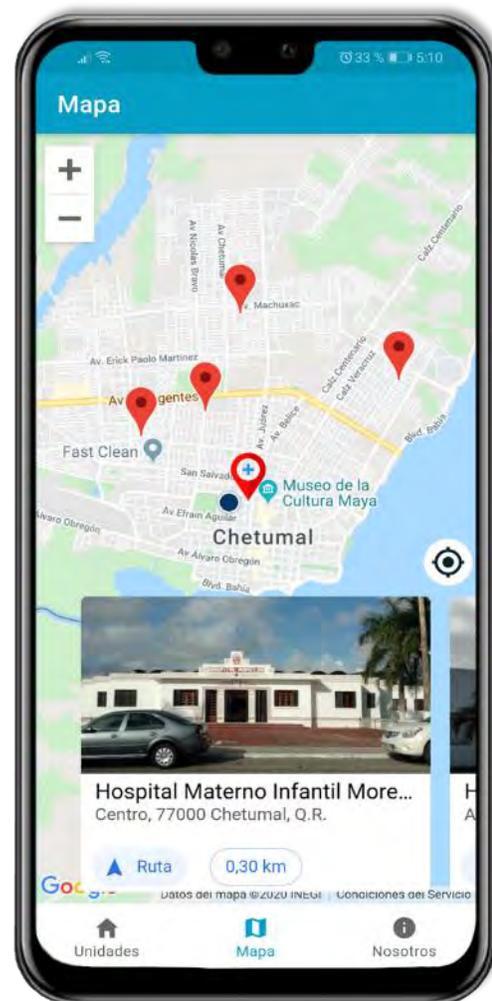


Figura 47. Pantalla de "Mapa", vista de tarjetas de información maximizada.

En la Figura 48 se observa la estructura básica para crear un marcador en un mapa. Los campos siguientes son particularmente importantes y se establecen comúnmente al construir un marcador:

- **Posición (obligatorio):** Especifica la ubicación inicial del marcador.
- **Mapa (opcional):** Especifica en que mapa se colocará el marcador. Si no especifica el mapa en la construcción del marcador, el marcador se crea, pero no se adjunta a ningún mapa (ni se muestra en él).

```
function initMap() {  
  var myLatLng = {lat: -25.363, lng: 131.044};  
  
  var map = new google.maps.Map(document.getElementById('map'), {  
    zoom: 4,  
    center: myLatLng  
  });  
  
  var marker = new google.maps.Marker({  
    position: myLatLng,  
    map: map,  
    title: 'Hello World!'  
  });  
}
```

Figura 48. Ejemplo para crear un marcador con la API de Google Maps. (Google, 2020)

Este proceso se realiza en un ciclo para crear los marcadores de cada una de las unidades mostradas actualmente en la pantalla de “unidades”, ya que como fue mencionado anteriormente estas dos pantallas se encuentran vinculadas, de tal manera que la vista de “mapa” siempre mostrará las mismas unidades que se encuentren actualmente en la vista de “unidades”, si el usuario aplico algún tipo de filtrado de unidades, este listado de unidades cambiara tanto como en la vista de unidades como en la de mapa.

Seguir el posicionamiento del usuario

Para esta funcionalidad se hace uso una vez más del plugin de geolocalización de Apache Cordova, que en conjunto con los marcadores de *Google Maps* muestran en el mapa el posicionamiento actual del usuario, con actualizaciones de estado cada 5 segundos.

En la Figura 49 se observa remarcado de color rojo el botón que debe tocar el usuario para iniciar el seguimiento de su posicionamiento geográfico. Una vez que el botón sea pulsado aparecerá en el mapa un nuevo marcador, el mapa se acercará y centrará en él, este punto está marcado de azul en la misma figura.

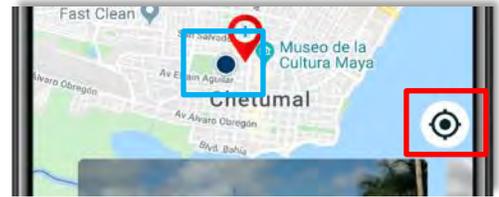


Figura 49. Botón para iniciar el rastreo y mostrar el marcador de posicionamiento.

Este botón iniciará una función que se ejecutará cada 5 segundos hasta que el usuario cambie de pantalla, cierre la aplicación móvil o desactive su GPS. La función se encargará de obtener el posicionamiento actual del usuario con la geolocalización de Apache Cordova y cambiará de ubicación el marcador de posicionamiento del usuario (punto azul).

Al realizar el cambio de posición en el marcador entre la posición anterior y la nueva posición se crea de manera manual una animación que muestra un desplazamiento del marcador entre las posiciones antes mencionadas, para realizar esta animación se crean varios marcadores intermedios y son mostrados cada uno de ellos en un periodo de tiempo corto.

3.3.6 Desarrollo de la funcionalidad de “Citas”

La función de agendar citas es presentada a través de una ventana emergente que puede ser llamada desde la pantalla de “unidades” y la pantalla “detalles”. Los botones que lanzan esta ventana emergente se pueden ver en la Figura 50 remarcados de color rojo.

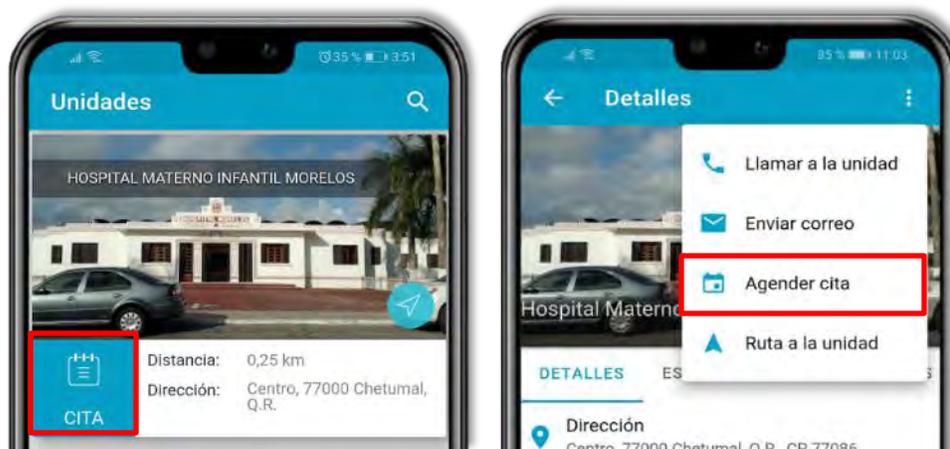


Figura 50. Botones para la funcionalidad de citas médicas.



Figura 51. Inicio de sesión para la funcionalidad de citas médicas.

Creación de cuentas de usuario

El formulario de registro se observa en la Figura 52 que es accesible al pulsar sobre la opción de “crear una cuenta nueva” desde la ventana de inicio de sesión (ver Figura 51). El usuario deberá completar todos los campos requeridos ingresando su CURP, correo electrónico y una contraseña, posteriormente al tocar sobre el botón de continuar se realizará una consulta a la API de RENAPO con la CURP que ingreso el usuario y se mostrará una alerta para confirmar su información personal. Ver Figura 53



Figura 52. Formulario de registro para la funcionalidad de citas médicas.



Figura 53. Información personal obtenida de RENAPO

Si la información es correcta al pulsar sobre el botón de aceptar se enviará el formulario de registro y pueden ocurrir dos situaciones (ver Figura 54):

- 1.- La cuenta se crea de manera exitosa: Se enviará un correo de confirmación al correo electrónico ingresado. La cuenta debe estar verificada para poder consultar la disponibilidad en la agenda médica de alguna unidad.
- 2.- La cuenta ya existe: Esta alerta aparece cuando el usuario está tratando de registrarse con un correo electrónico o una CURP que ya se encuentra asociada a un usuario en el sistema, el usuario deberá iniciar el proceso de recuperación de su cuenta pulsando sobre la opción “recuperar contraseña” desde la vista de inicio de sesión (ver Figura 51).

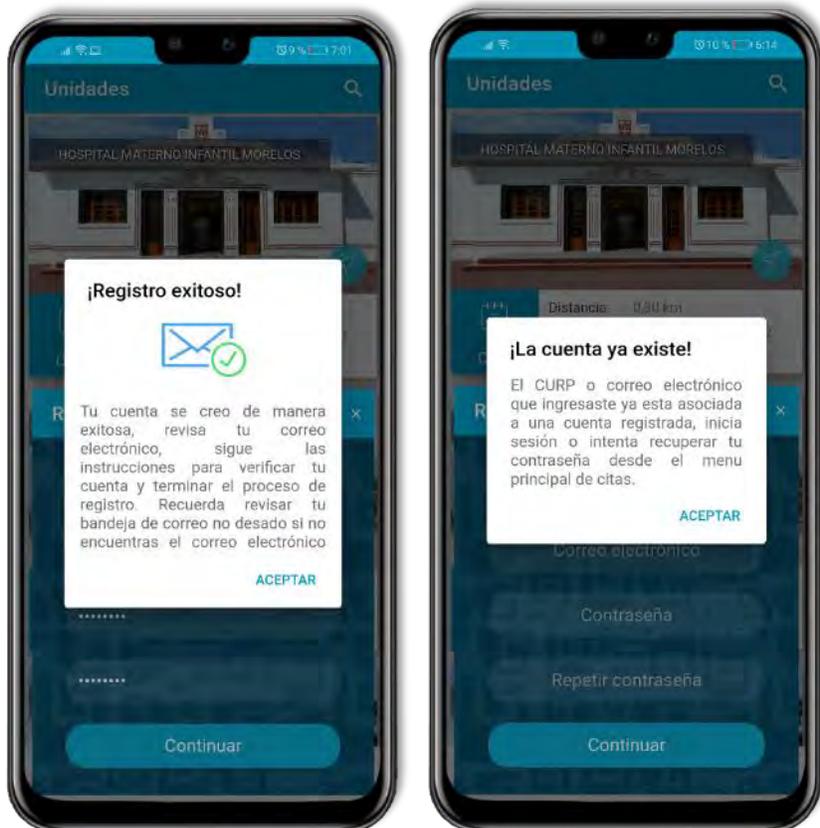


Figura 54. Posibles mensajes de alerta al crear una cuenta en la aplicación móvil.

Agenda Médica

Una vez que el usuario ha iniciado sesión exitosamente la ventana ahora mostrará un calendario para seleccionar un día, también desde esta nueva ventana el usuario puede cerrar su sesión (Ver Figura 55). Cuando el usuario toca sobre un día se realizará una petición al back-end para solicitar el horario de atención que brinda la unidad de salud en el día seleccionado, al procesar esta solicitud pueden suceder hasta 4 diferentes situaciones:

1.- El usuario ya cuenta con una cita médica agendada: Esta alerta aparecerá cuando el usuario cuenta con una cita activa, es decir, que no ha sido marcada como atendida o aún no ha expirado, desde esta alerta se puede reenviar nuevamente al correo electrónico del usuario todos los detalles de la cita activa o cancelarla. Ver en la Figura 56



Figura 55. Calendario para consultar la agenda médica de una unidad de salud.

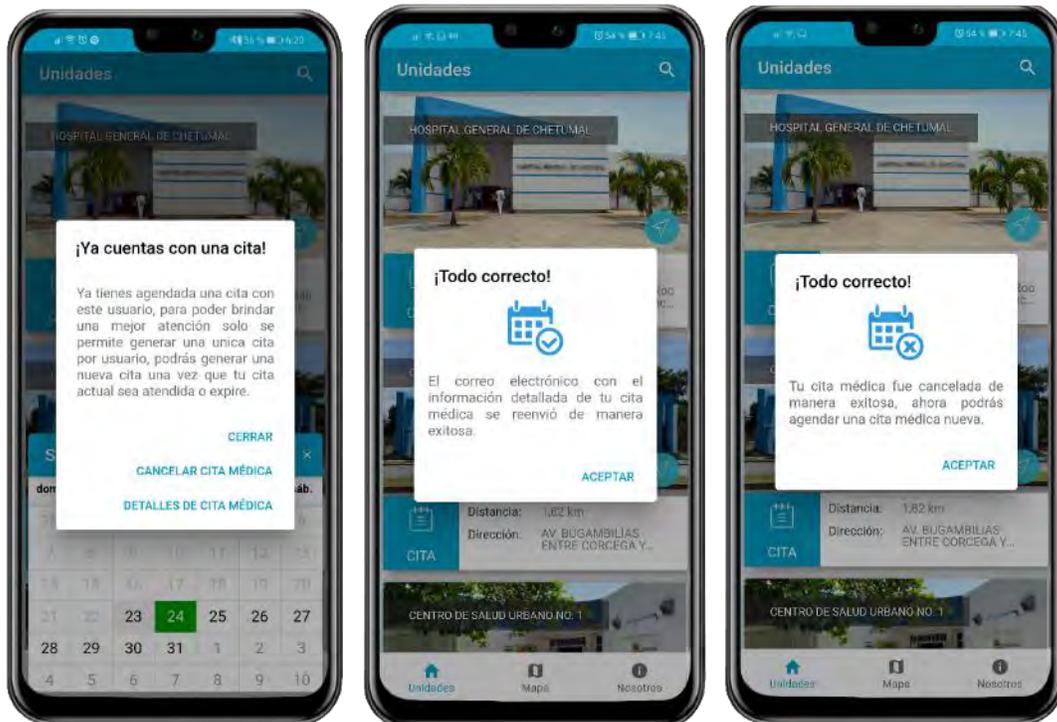


Figura 56. Alertas para un usuario que ya cuenta con una cita agendada.

2.- Sin servicio disponible: Este error aparece cuando la unidad de salud no tiene dado de alta el servicio de consulta externa, el usuario deberá seleccionar una unidad diferente para poder agendar una cita médica.

3.- Sin citas disponibles: Este error aparece cuando la unidad ya no cuenta con citas para el día seleccionado, este mensaje también puede aparecer cuando el usuario está intentando agendar una cita el mismo día fuera del horario de atención del día seleccionado.

4.- Selección de horario: Cuando la unidad de salud tiene citas disponibles para el día seleccionado se cambiará la información mostrada en el modal por un listado de horarios disponibles con los que cuenta la unidad médica para agendar una cita.

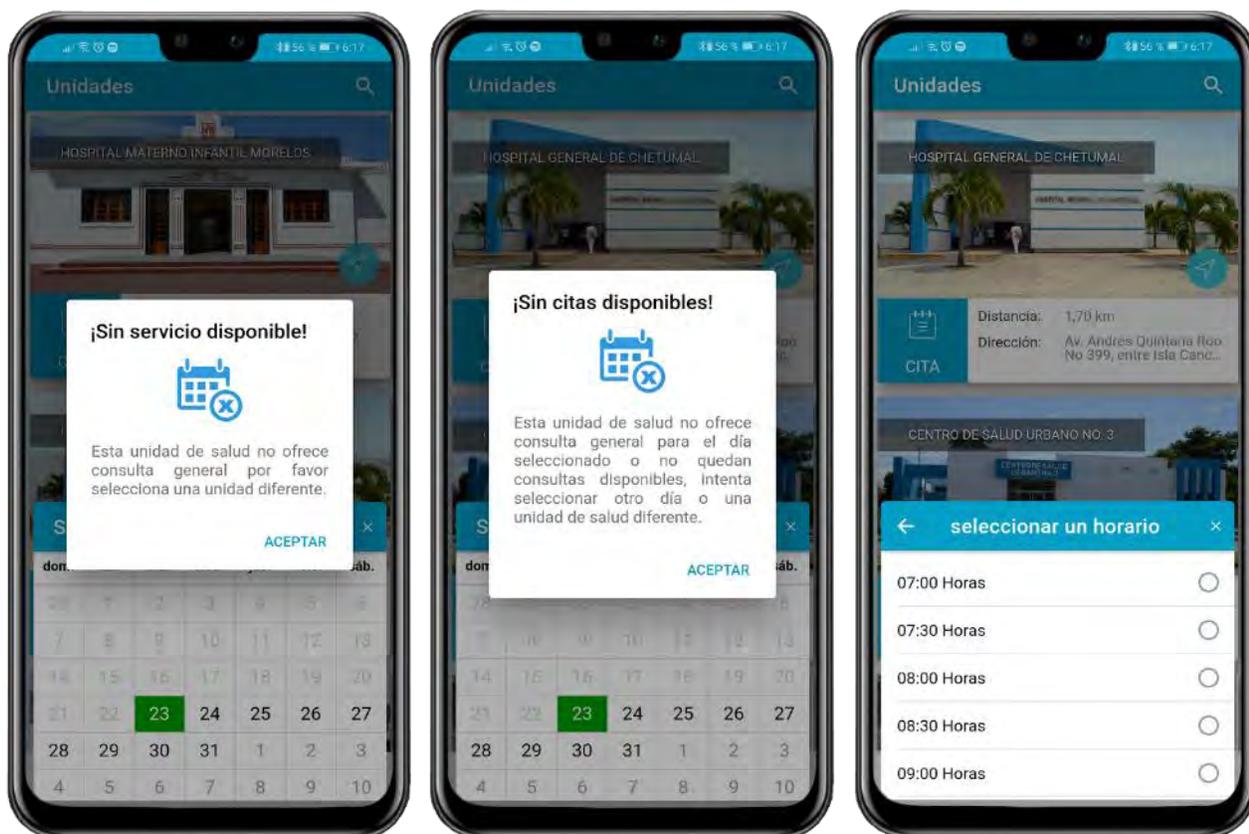


Figura 57. Posibles respuestas al consultar la agenda médica de una unidad.

Finalmente, cuando se pulse sobre algún horario disponible del listado se lanzará una alerta de confirmación con el propósito que el usuario valide que los datos de la cita que esta por agendar sean correctos, el mensaje muestra la unidad de salud, fecha y hora seleccionada.

Si el usuario pulsa sobre la opción de cancelar se cerrará la alerta y se podrá seleccionar otro horario disponible, si todos los datos son correctos y se confirma la creación de la cita médica el sistema realizará una petición al back-end para almacenar la cita, una vez que se devuelva una respuesta exitosa será mostrada la alerta que notifica la creación exitosa de la cita, de manera simultanea se envía un correo electrónico a la cuenta que haya registrado el usuario con todos los detalles de la cita que fue agendada. Ver Figura 58.

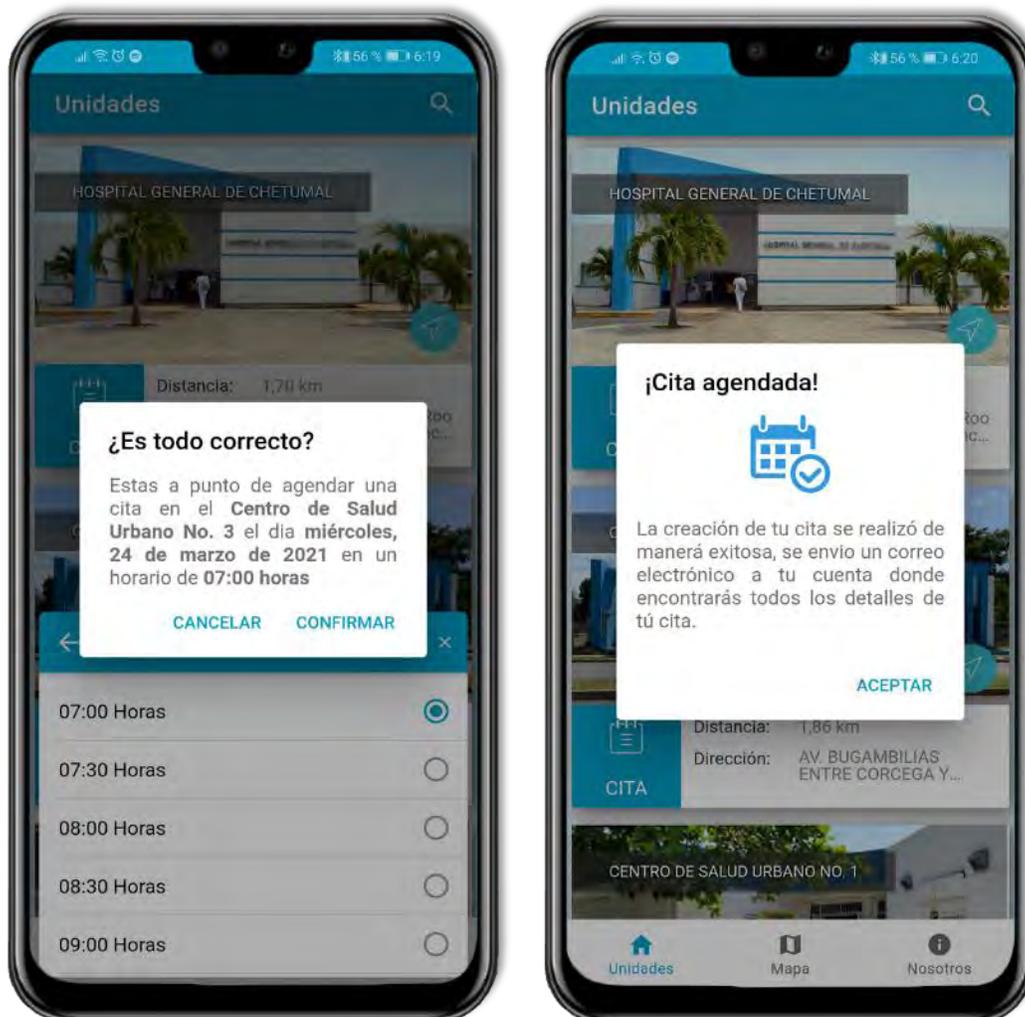


Figura 58. Mensaje de confirmación para agendar una cita médica.

Reestablecer contraseña



Figura 59. Solicitud para reestablecer contraseña.

Una vez que el usuario ingrese su CURP y pulse sobre la opción de enviar, se realizará la petición web hacia el back-end, este se encargará de procesar la solicitud para validar la CURP ingresada y enviar a la dirección de correo electrónico asociada el enlace para restablecer la contraseña, una vez que la petición termine de ser procesada y el enlace sea enviado se mostrará una notificación en la aplicación móvil (ver Figura 60).

La opción para reestablecer contraseña es accesible desde la vista de inicio de sesión para citas médicas (ver Figura 59) y permite al usuario recuperar el acceso a su cuenta actualizando su contraseña por una nueva.

Al tocar sobre la opción de “recuperar contraseña” será mostrada una ventada que solicita la CURP que ingreso el usuario al momento de la creación de su cuenta como se ve en la Figura 59.



Figura 60. Notificación de correo enviado exitosamente para reestablecer contraseña.

3.4 Fase de Pruebas

3.4.1 Elaboración del plan de pruebas

Las pruebas son un conjunto de actividades que se desarrollan con el objetivo de verificar la correcta funcionalidad del software que fue programado, permiten detectar la existencia o ausencia de errores y comprobar el desempeño de la aplicación móvil.

En esta sección se enlistan cada una de las pruebas realizadas en la aplicación, se encuentran divididas en tres tipos de pruebas; Pruebas unitarias que permiten evaluar individualmente las funcionalidades en el software desarrollado, pruebas de integración que permiten validar el consumo de las *APIs* empleadas y pruebas de interfaces que permiten validar el diseño gráfico de las pantallas creadas.

El cumplimiento de todas las pruebas descritas a continuación fue llevado a cabo por el equipo de desarrollo en conjunto con el Coordinador de Informática (*stakeholder*), responsable del proyecto, dado que la aplicación móvil no ha sido liberada para el público en general todas las pruebas realizadas fueron con personal de la Coordinación de Informática de los Servicios Estatales de Salud y validadas por el responsable del proyecto.

Pruebas unitarias*Tabla 1. Listado de pruebas unitarias.*

No.	Prueba	Cumple
1	Lanzamiento de la pantalla de bienvenida.	Si
2	Listar las unidades de salud que se encuentren en la localidad del usuario.	Si
3	Cálculo de la distancia entre las unidades de salud y el usuario.	Si
4	Ordenar el listado de unidades de salud por cercanía.	Si
5	Paginar los resultados mostrados en el listado.	Si
6	Actualizar la información de las unidades de salud al refrescar la pantalla.	Si
7	Consultar la información detallada de las unidades de salud.	Si
8	Lanzamiento de la aplicación nativa de llamadas con el número pre marcado de la unidad de salud seleccionada.	Si
9	Lanzamiento de la aplicación nativa de correo electrónico con un nuevo correo con destinatario a la unidad de salud seleccionada.	Si
10	Visualización de la ubicación de la unidad de salud en la vista de detalles a través de un mapa estático.	Si
11	Visualización en pantalla completa de las imágenes de las unidades de salud.	Si
12	Filtrar el listado de unidades de salud.	Si
13	Visualización de la pantalla de carga de contenido.	Si
14	Visualización de los marcadores para las unidades de salud en la pantalla de mapas.	Si
15	Visualización de las tarjetas de información en la pantalla de mapas.	Si
16	Centrar el mapa en la localidad del usuario.	Si
17	Seguimiento del posicionamiento geográfico del usuario en el mapa.	Si
18	Lanzamiento de la aplicación nativa de mapas con ruta hacia una unidad de salud seleccionada.	Si
19	Registro de cuentas de usuario.	Si
20	Inicio de sesión con cuentas de usuario.	Si
21	Reestablecer contraseñas.	Si
22	Consultar la disponibilidad en la agenda médica de la unidad de salud.	Si
23	Agendar una cita médica.	Si

Pruebas de integración

Tabla 2. Pruebas de integración con el back-end de la aplicación móvil.

Back-end de la aplicación móvil		
No.	Prueba	Cumple
1	Autenticación de la aplicación móvil.	Si
2	Generar token para peticiones web.	Si
3	Consulta de unidades de salud.	Si
4	Consulta de detalles de una unidad de salud.	Si
5	Almacenamiento de citas médicas.	Si
6	Administración de citas médicas de las unidades de salud.	Si
7	Administración de cuentas de usuario para citas médicas.	Si

Tabla 3. Pruebas de integración con el Web Service de Google Maps.

Web Service Google Maps		
No.	Prueba	Cumple
1	Consulta de Place ID del usuario.	Si
2	Generación de mapas estáticos.	Si
3	Generación de mapas dinámicos.	Si

Tabla 4. Pruebas de integración con la API de RENAPO.

Web Service RENAPO		
No.	Prueba	Cumple
1	Consulta de información personal a través de la CURP.	Si

Pruebas de interfaces

Tabla 5. Listado de pruebas de interfaces.

No.	Prueba	Cumple
1	El diseño de la pantalla cumple con el diseño acordado.	Si
2	Cada función es llamada correctamente al pulsar sobre el elemento gráfico al que se encuentren asociado en la pantalla.	Si
3	Los enlaces a otras pantallas funcionan de acuerdo con el diagrama de navegación.	Si
4	Los elementos gráficos no se superponen entre ellos.	Si
5	La relación de aspecto se mantiene entre dispositivos móviles.	Si
6	Las animaciones son renderizadas de manera correcta.	Si

3.4.2 Ejecución de las pruebas

En esta sección se describen las condiciones que se deben tener en cuenta al realizar las pruebas, el proceso de ejecución, los resultados esperados y finalmente los resultados obtenidos. Las pruebas listadas anteriormente se agruparon por funcionalidades principales de la aplicación móvil, cada tabla mostrada a continuación muestra los resultados obtenidos por el conjunto de pruebas unitarias incluidos en la funcionalidad desarrollada.

Tabla 6. Ejecución de la prueba listar unidades de salud.

Listar unidades de salud	
Descripción	La funcionalidad principal de la pantalla de “Unidades” es listar las unidades de salud ordenadas por cercanía y que se encuentren en la localidad del usuario al iniciar la aplicación móvil.
Precondición	Contar con una conexión a internet.
Proceso de ejecución	
<ul style="list-style-type: none"> • Una vez cumplidas las precondiciones el usuario debe ejecutar la aplicación móvil. • Esperar que la pantalla de bienvenida sea removida y aceptar los permisos solicitados por la aplicación móvil para acceder al GPS. 	
Resultado esperado	
Caso 1 El usuario acepta los permisos solicitados por la aplicación.	Se lista satisfactoriamente las unidades de salud ordenadas por cercanía y que se encuentren en la localidad del usuario.
Caso 2 El usuario rechaza los permisos solicitados por la aplicación.	La aplicación redirige al usuario a la pantalla de filtros para que el usuario busque manualmente unidades de salud a través de los filtros de búsqueda.
Evaluación	Prueba aprobada exitosamente.

La ejecución de las pruebas de funcionalidad restantes de la aplicación móvil se encuentra en el **Anexo 5. Ejecución de pruebas**

3.5 Documentación

La documentación requerida en la implementación de la aplicación móvil está conformada por la *configuración del entorno de desarrollo* (ver Anexo 1. Configuración del entorno de desarrollo), que tiene como objetivo orientar al personal encargado de realizar cualquier actualización en la aplicación y por la *infografía de la aplicación móvil* (Ver Anexo 2. Infografía de la aplicación móvil.), que tiene como objetivo promocionar la aplicación y dar a conocer las principales características de esta al público en general una vez liberada.

Infografía de la aplicación móvil: Una infografía es una representación visual de datos de forma rápida y clara para el público, para el caso de la aplicación móvil la infografía destaca las funciones más importantes que brinda la aplicación y además permite una difusión de mayor impacto.

Configuración del entorno de desarrollo: Se detallan las configuraciones de cada una de las herramientas que son utilizadas para realizar la implementación del entorno de desarrollo de la aplicación móvil, en caso de que se deban realizar procesos externos a cada una de las herramientas también serán especificados.

Capítulo 4 Conclusiones

El proyecto realizado responde a las expectativas y requerimientos que se establecieron al inicio de este, en el ciclo de desarrollo que lleva por nombre *plan de lanzamiento* de acuerdo con la metodología *Extreme Programming*. Pese a que, a causa de las limitaciones existentes aún, este debe ser regulado para poder ser distribuido públicamente, lo cierto es que el alcance de este proyecto y sus objetivos fueron cubiertos en su totalidad.

El desarrollo de la aplicación móvil hace posible a cualquier persona obtener información de las unidades de salud a su alrededor y de cualquier unidad de SESA en el Estado de Quintana Roo, el servicio de citas médicas desde la aplicación móvil brinda a la población en general la facilidad de agendar una cita sin la necesidad de acudir a la unidad médica sino hasta el día agendado, en consecuencia, se facilita el acceso a la información y al servicio de citas médicas.

Todas las fases de desarrollo se llevaron a cabo de manera exitosa y tal como lo indica la metodología propuesta, gracias a esta metodología ágil de desarrollo de software y las iteraciones de desarrollo que propone fue posible presentar entregables funcionales a los interesados en el proyecto al finalizar cada ciclo de desarrollo hasta concluir con los módulos planificados.

El uso de *Ionic* en este proyecto fue un punto clave para la gestión del tiempo de desarrollo ya que al usar tecnologías web se agiliza la curva de aprendizaje en comparación con el uso de un lenguaje de desarrollo para construir aplicaciones móviles nativas, a través de la metodología seleccionada se privilegiaron las buenas prácticas de programación hasta obtener el producto final aprobado por el cliente.

4.1 Trabajo a futuro

El desarrollo de este proyecto tuvo en la mira a otro proyecto del área de la salud: el **Expediente Clínico Electrónico** que utiliza los Servicios Estatales de Salud en Quintana Roo. La aplicación móvil podría obtener la agenda médica de las unidades de salud desde este sistema, el expediente se encargaría de la gestión de citas médicas e incluso se podría obtener información detallada de las unidades de salud desde este sistema, todo a través del consumo de *APIs*. El expediente actualmente maneja toda esta información, pero por diversas restricciones no se permitió el acceso de este sistema para la comunicación con la aplicación móvil, sin embargo, se contempló para su uso futuro.

En el servidor de la aplicación, se puede incluir también el desarrollo de una versión web con todas las funcionalidades de la aplicación móvil. Al ser una aplicación híbrida desarrollada con tecnologías web, la implementación de una versión web no requeriría mucha adaptación en el código base, actualmente existe un sistema web únicamente para la administración de la aplicación, en este mismo sitio se puede incluir una parte pública que contendría la aplicación web para brindar al usuario final el acceso web con funcionalidades similares a la aplicación móvil. Gracias a la versatilidad y el diseño modular que nos brindan las tecnologías empleadas en el desarrollo, se podría agregar un menú o más pestañas de navegación para brindar un catálogo más amplio de servicios e información disponible en la aplicación móvil.

Dado el amplio abanico de posibilidades que ofrece el framework de desarrollo y las tecnologías web se pueden hacer diversas mejoras visuales y posiblemente de rendimiento, cada mejora podría publicarse como actualizaciones de versión una vez que se encuentre en la tienda de aplicaciones, sin embargo, todo dependerá de la funcionalidad que se necesite mejorar y del desempeño que presente la versión actual de la aplicación móvil.

Referencias

- ANGULAR JS. (2020). *The Basics*. Retrieved from angularjs: <https://angularjs.org/>
- Cabello, M. V. (2010). *Introducción a las bases de datos relacionales*. Madrid: Vision Libros.
- Danilo Geovanny Barreno Naranjo, D. P. (2016). *Contribuciones a las Ciencias Sociales*. Retrieved from <http://www.eumed.net/rev/cccss/2016/04/5G.html>
- ENDUTIH. (2018). *Encuesta Nacioal de Disponibilidad y Uso de Tecnologías de la Información en los Hogares*. Retrieved from <http://>
- González, J. A. (2017). *Sistema de información para películas con angularJS*.
- Google. (2020). *Developer Guide*. Retrieved from Google Maps Platform: <https://developers.google.com/maps/documentation/geocoding/intro>
- Google. (2020). *Precios para maps, routesy places*. Retrieved from Google Cloud: <https://cloud.google.com/maps-platform/pricing/sheet?hl=es-419>
- Griffith, C. (2017). *Mobile App Development with Ionic, Revised Edition*. Sebastapol: O'Reilly Media, Inc.
- Hernando Rábanos, J., Mendo Tomás, L., & Riera Salís, J. M. (2015). *Comunicaciones móviles*. Madrid: Editorial Universitaria Ramón Areces.
- IMSS. (2016, diciembre 29). *Aplicación para dispositivos móviles IMSS Digital*. Retrieved from Acciones y Programas: <https://www.gob.mx/imss/acciones-y-programas/app-imss-digital>
- INEGI. (2017). *Anuario estadístico y geográfico de Quintana Roo 2017*.
- Ionic. (2020). *Geolocation Documentation*. Retrieved from Ionic Framework: <https://ionicframework.com/docs/native/geolocation>
- Ionic. (2020). *Ionic Framework*. Retrieved from <https://ionicframework.com/>
- Ionic. (2020). *Ion-Refresher Documentation*. Retrieved from Ionic Framework: <https://ionicframework.com/docs/api/refresher>
- Ionic. (2021). *Android Play Store Deployment*. Retrieved from Ionic Framework: <https://ionicframework.com/docs/deployment/play-store>
- Joskowicz, J. (2008). *Reglas y Prácticas en eXtreme Programming*.

- Matrosov, N., Kokin, V., & Tyurina, S. (2017). Apache Cordova. PROBLEMAS DE USO EFECTIVO DEL POTENCIAL CIENTÍFICO DE LA EMPRESA. *OMEGA SCIENCE*, 74-75.
- OMS. (2016). *mSalud: uso de las tecnologías móviles inalámbricas en la salud pública*. Organización Mundial de la Salud.
- Ruiz de Chavez, M., & Martínez Narváez, G. (1988). *EL PAPEL DE LA JURISDICCIÓN SANITARIA EN LOS SISTEMAS ESTATALES DE SALUD*. México, D.F.
- Secretaria de Salud del Estado de Quintana Roo*. (n.d.). Retrieved Marzo 21 , 2019, from <https://qroo.gob.mx/sesa/nuestra-historia>
- Secretaria de Salud del Estado de Quintana Roo*. (n.d.). Retrieved Marzo 21, 2019, from <https://qroo.gob.mx/sesa/mision-y-vision>
- Secretaria de Salud del Estado de Quintana Roo*. (2010, Marzo). Retrieved Marzo 25, 2019, from <https://salud.qroo.gob.mx/portal/descargas/manual1.pdf>
- SESA. (2019). *mapa de unidades*. Retrieved from Servicios Estatales de salud: <https://www.qroo.gob.mx/sesa/mapa-unidades>
- SESA. (2020, Julio 20). *ORGANIGRAMA*. Retrieved from <https://qroo.gob.mx/sesa/organigrama>

Anexos

Anexo 1. Configuración del entorno de desarrollo

El siguiente anexo muestra los procesos que debe seguir un usuario para poner en marcha la aplicación móvil en un entorno de desarrollo y generar una versión de lanzamiento de la aplicación que puede ser cargada a la Play Store.

Descarga del proyecto

Para iniciar con la ejecución de la aplicación móvil en un entorno de pruebas el primer paso consiste en clonar la última versión del proyecto desde el repositorio en el que se encuentra alojado. Para este paso se debe de tener instalado la aplicación *Git* y abrir una nueva terminal en el directorio donde será clonado este proyecto.

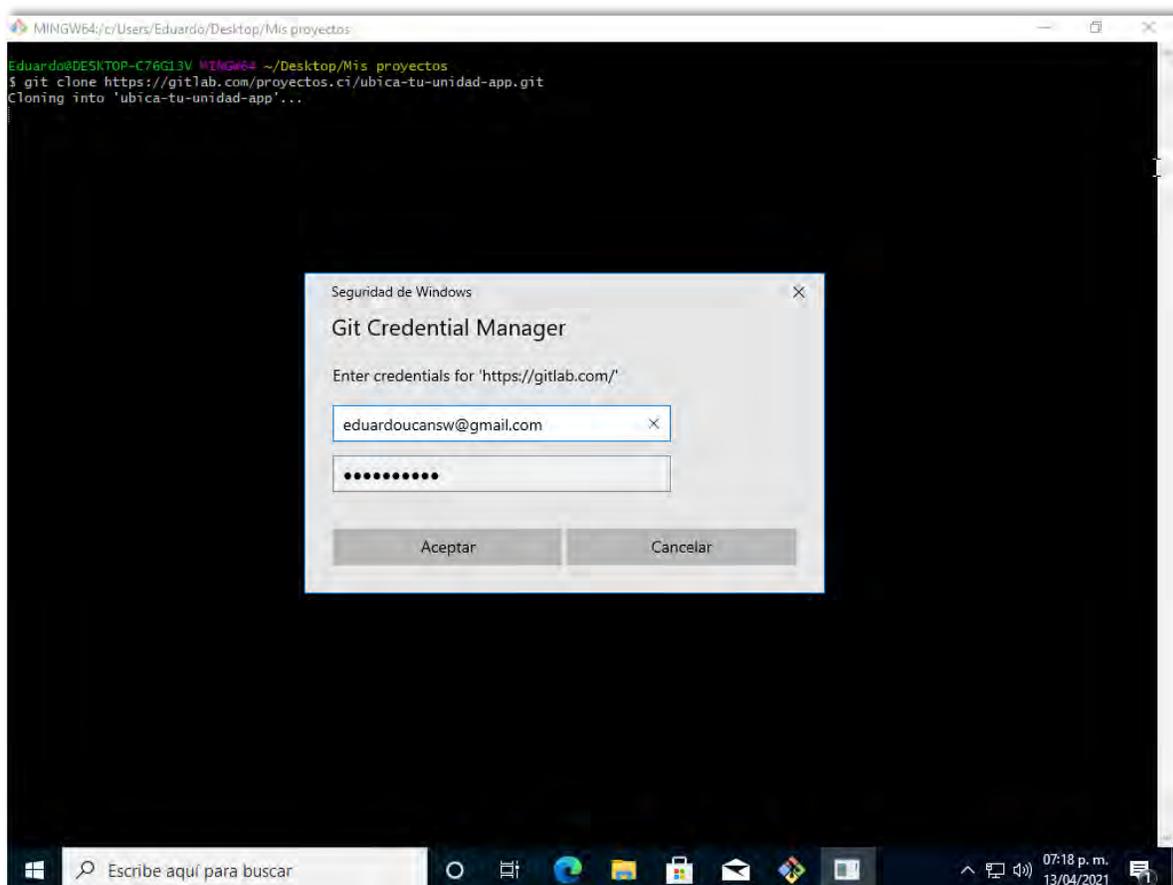
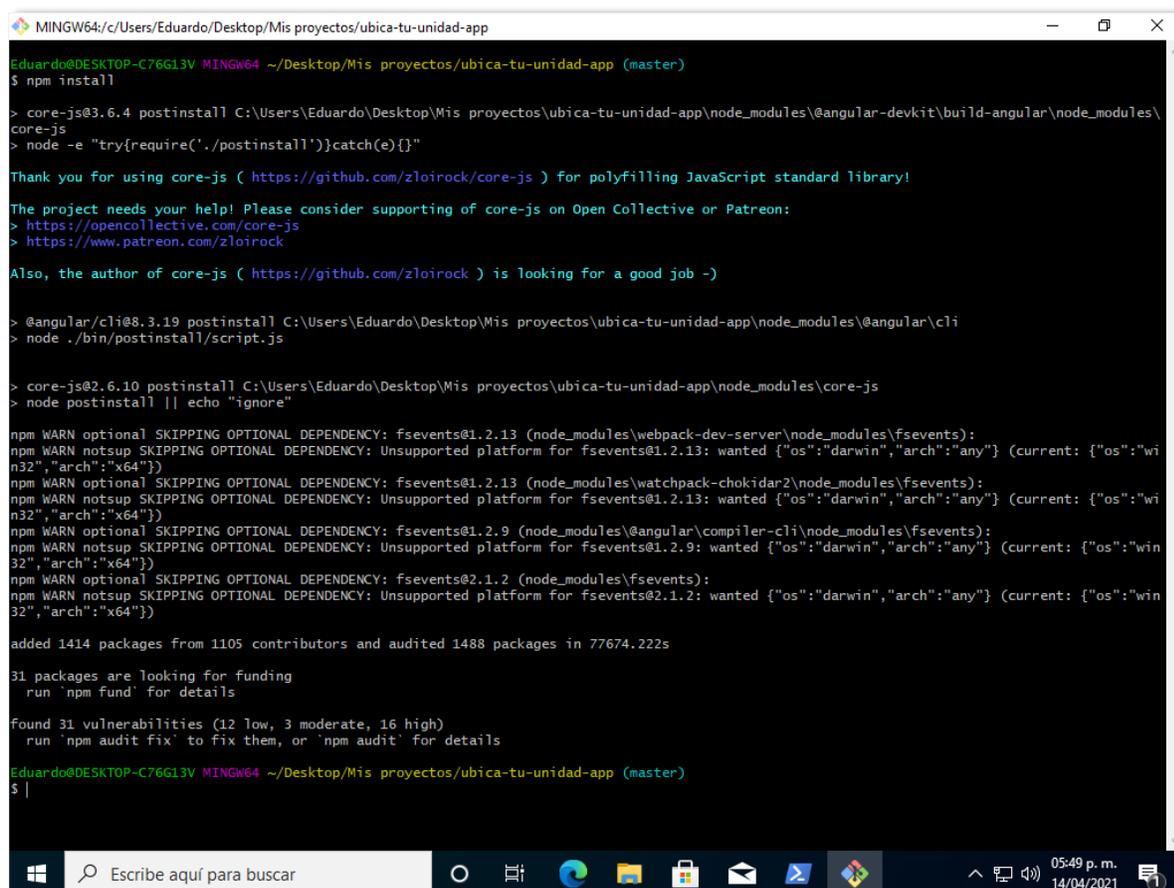


Figura 61. Clonación del proyecto.

Como se observa en la Figura 61, se ingresa el comando de clonación de Git, posterior a esta acción la aplicación abrirá una ventana emergente de autenticación antes de continuar con la descarga del proyecto, el usuario autenticado debe tener los permisos necesarios en el repositorio para clonar el proyecto.

Instalación de dependencias

Una vez finalizada la clonación del proyecto es necesario instalar todas las dependencias de la aplicación móvil a través del sistema de gestión de paquetes por defecto para **Node JS** ejecutando el comando “`npm install`” (Figura 62), para este paso es necesario tener instalado Node JS.



```
MINGW64/c:/Users/Eduardo/Desktop/Mis proyectos/ubica-tu-unidad-app
Eduardo@DESKTOP-C76G13V MINGW64 ~/Desktop/Mis proyectos/ubica-tu-unidad-app (master)
$ npm install

> core-js@3.6.4 postinstall C:\Users\Eduardo\Desktop\Mis proyectos\ubica-tu-unidad-app\node_modules\@angular-devkit\build-angular\node_modules\
core-js
> node -e "try{require('./postinstall')}catch(e){}"

Thank you for using core-js ( https://github.com/zloirock/core-js ) for polyfilling JavaScript standard library!

The project needs your help! Please consider supporting of core-js on Open Collective or Patreon:
> https://opencollective.com/core-js
> https://www.patreon.com/zloirock

Also, the author of core-js ( https://github.com/zloirock ) is looking for a good job :-)
```

```
> @angular/cli@8.3.19 postinstall C:\Users\Eduardo\Desktop\Mis proyectos\ubica-tu-unidad-app\node_modules\@angular\cli
> node ./bin/postinstall/script.js

> core-js@2.6.10 postinstall C:\Users\Eduardo\Desktop\Mis proyectos\ubica-tu-unidad-app\node_modules\core-js
> node postinstall || echo "ignore"
```

```
npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@1.2.13 (node_modules\webpack-dev-server\node_modules\fsevents):
npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for fsevents@1.2.13: wanted {"os":"darwin","arch":"any"} (current: {"os":"win
n32","arch":"x64"})
npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@1.2.13 (node_modules\watchpack-chokidar2\node_modules\fsevents):
npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for fsevents@1.2.13: wanted {"os":"darwin","arch":"any"} (current: {"os":"win
n32","arch":"x64"})
npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@1.2.9 (node_modules\@angular\compiler-cli\node_modules\fsevents):
npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for fsevents@1.2.9: wanted {"os":"darwin","arch":"any"} (current: {"os":"win
32","arch":"x64"})
npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@2.1.2 (node_modules\fsevents):
npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for fsevents@2.1.2: wanted {"os":"darwin","arch":"any"} (current: {"os":"win
32","arch":"x64"})

added 1414 packages from 1105 contributors and audited 1488 packages in 77674.222s

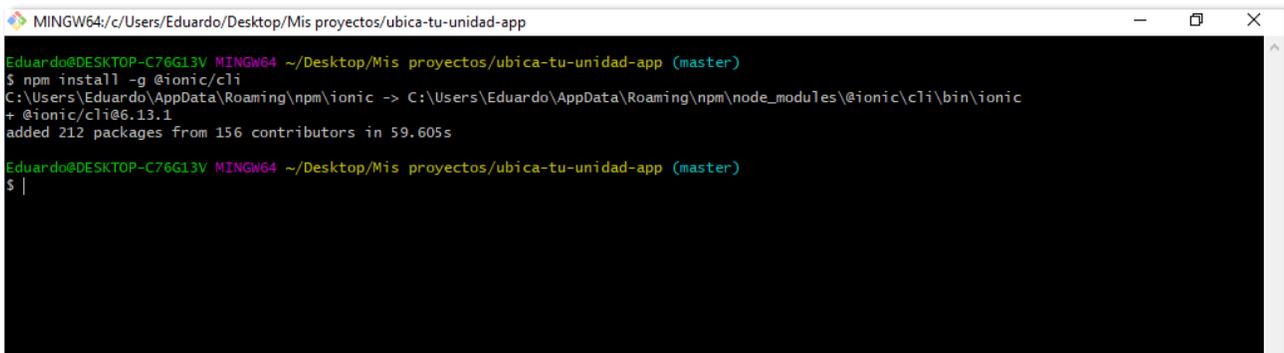
31 packages are looking for funding
  run 'npm fund' for details

found 31 vulnerabilities (12 low, 3 moderate, 16 high)
  run 'npm audit fix' to fix them, or 'npm audit' for details

Eduardo@DESKTOP-C76G13V MINGW64 ~/Desktop/Mis proyectos/ubica-tu-unidad-app (master)
$ |
```

Figura 62. Instalación de dependencias del proyecto.

Para continuar con la ejecución del proyecto en un entorno de desarrollo instalamos la interfaz de línea de comandos **Ionic CLI** ejecutando el comando “`npm install -g @Ionic/cli`”. Ver Figura 63



```
MINGW64:/c/Users/Eduardo/Desktop/Mis proyectos/ubica-tu-unidad-app
Eduardo@DESKTOP-C76G13V MINGW64 ~/Desktop/Mis proyectos/ubica-tu-unidad-app (master)
$ npm install -g @ionic/cli
C:\Users\Eduardo\AppData\Roaming\npm\ionic -> C:\Users\Eduardo\AppData\Roaming\npm\node_modules\@ionic\cli\bin\ionic
+ @ionic/cli@6.13.1
added 212 packages from 156 contributors in 59.605s
Eduardo@DESKTOP-C76G13V MINGW64 ~/Desktop/Mis proyectos/ubica-tu-unidad-app (master)
$
```

Figura 63. Instalación de Ionic CLI.

Finalmente, para iniciar el proyecto se ingresa el comando “*Ionic serve*”, una vez que el proceso de ejecución finalice aparecerá un mensaje en la terminal confirmando la ejecución del proyecto y se abrirá una ventana en el navegador con la aplicación móvil en la dirección local del equipo en el puerto 8100 (<http://localhost:8100>). Ver Figura 64.

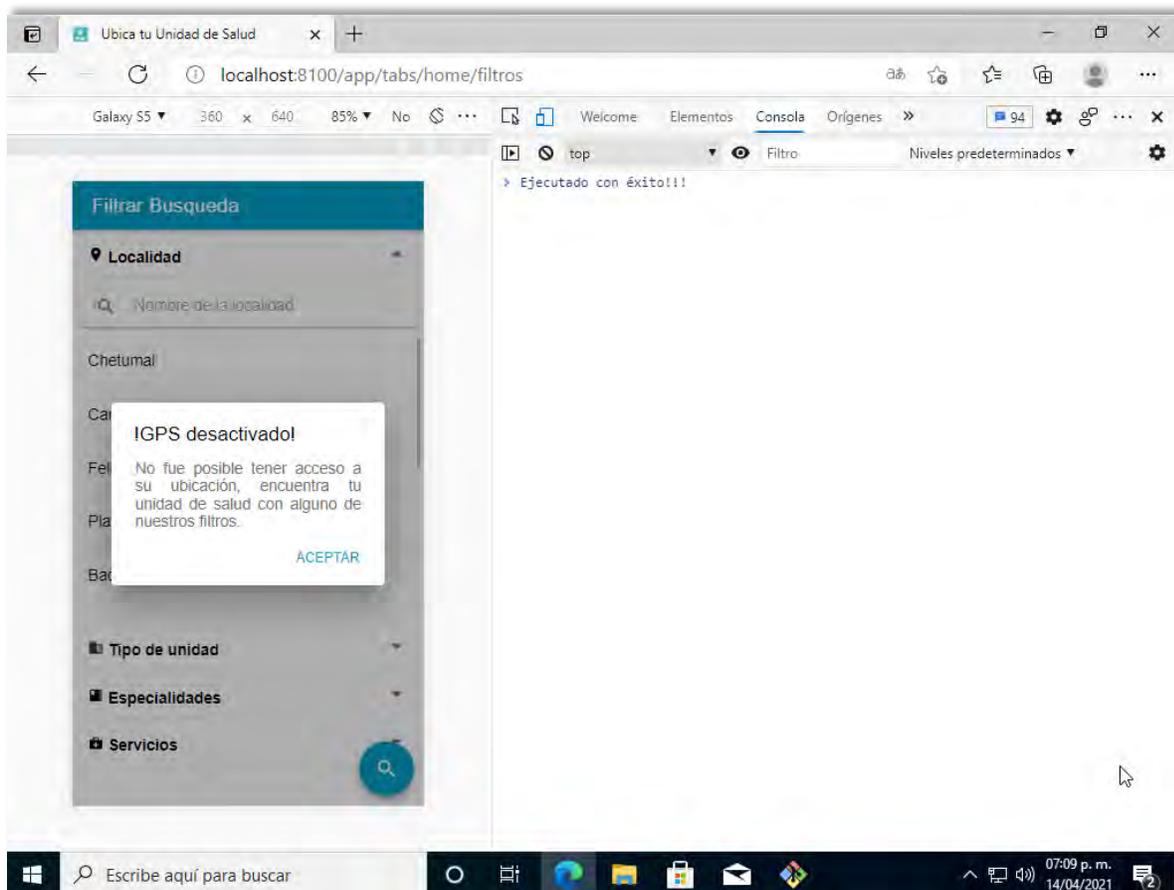


Figura 64. Aplicación móvil corriendo en un entorno local de desarrollo.

Despliegue de la aplicación móvil

Para el despliegue de la aplicación seguimos los pasos marcados por la documentación oficial de *Ionic* (Ionic, 2021) de acuerdo con la plataforma en la que se desea generar la versión de lanzamiento. En esta sección se mostrarán los pasos necesarios para generar el paquete de aplicación Android (APK) y los pasos para subir el aplicativo a la Play Store una vez que el desarrollador haya concluido las actualizaciones en la aplicación móvil.

Configurar el entorno de desarrollo de Android

Para iniciar la configuración es necesario descargar la última versión de Android Studio desde el sitio oficial, de acuerdo con el sistema operativo en el que se desea generar el paquete de Android. Una vez que la descarga finalice se ejecuta el instalador y se siguen todos los pasos de este hasta concluir con la instalación de la aplicación.

Al finalizar la instalación es necesario instalar la herramienta de línea de comandos de Android SDK marcando la opción “Android SDK Command-line Tools (latest)” en la sección Android SDK de la configuración del sistema en la aplicación, posterior a este paso es necesario hacer clic sobre el botón “Apply” para concluir la configuración. Ver Figura 65.

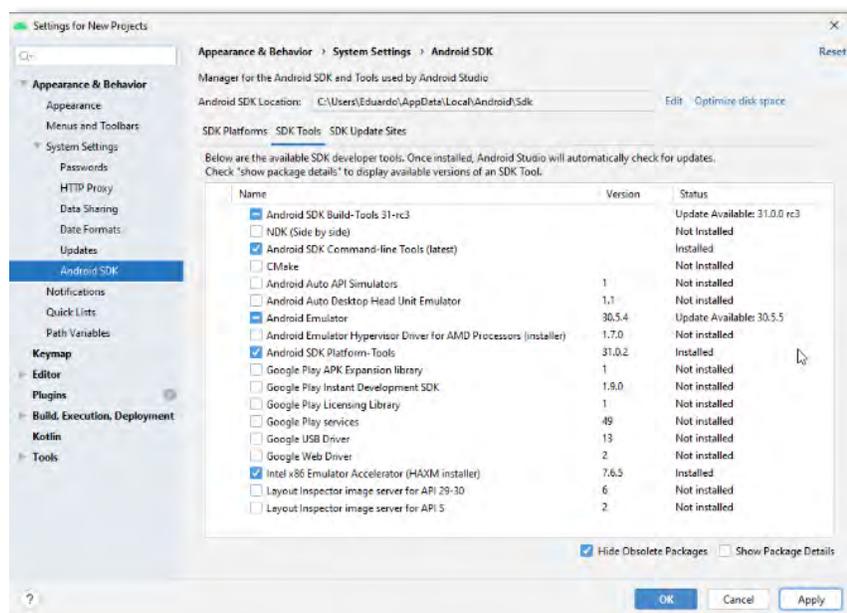


Figura 65. Instalación de la herramienta de línea de comandos Android SDK.

Una vez concluida la configuración de Android Studio instalamos *Gradle* en el equipo, Gradle es una herramienta de automatización de compilación, controla el proceso de desarrollo en las tareas de compilación y empaquetado para pruebas, implementación y publicación.

Para iniciar la instalación se descarga Gradle desde el sitio oficial y posteriormente se copian todos los archivos al directorio en el que se realizará la instalación. Finalizado el paso anterior es necesario agregar la ruta de Gradle a las variables del sistema, en Windows abrimos el editor de variables de entorno del sistema y hacemos clic sobre “nueva variable” agregamos el nombre y ruta de Gradle como se ve en la Figura 66 y se da clic en aceptar, editamos la variable del sistema que lleva por nombre “path” para agregar la ruta a Gradle (ver Figura 67), al finalizar se guardan todos los cambios.

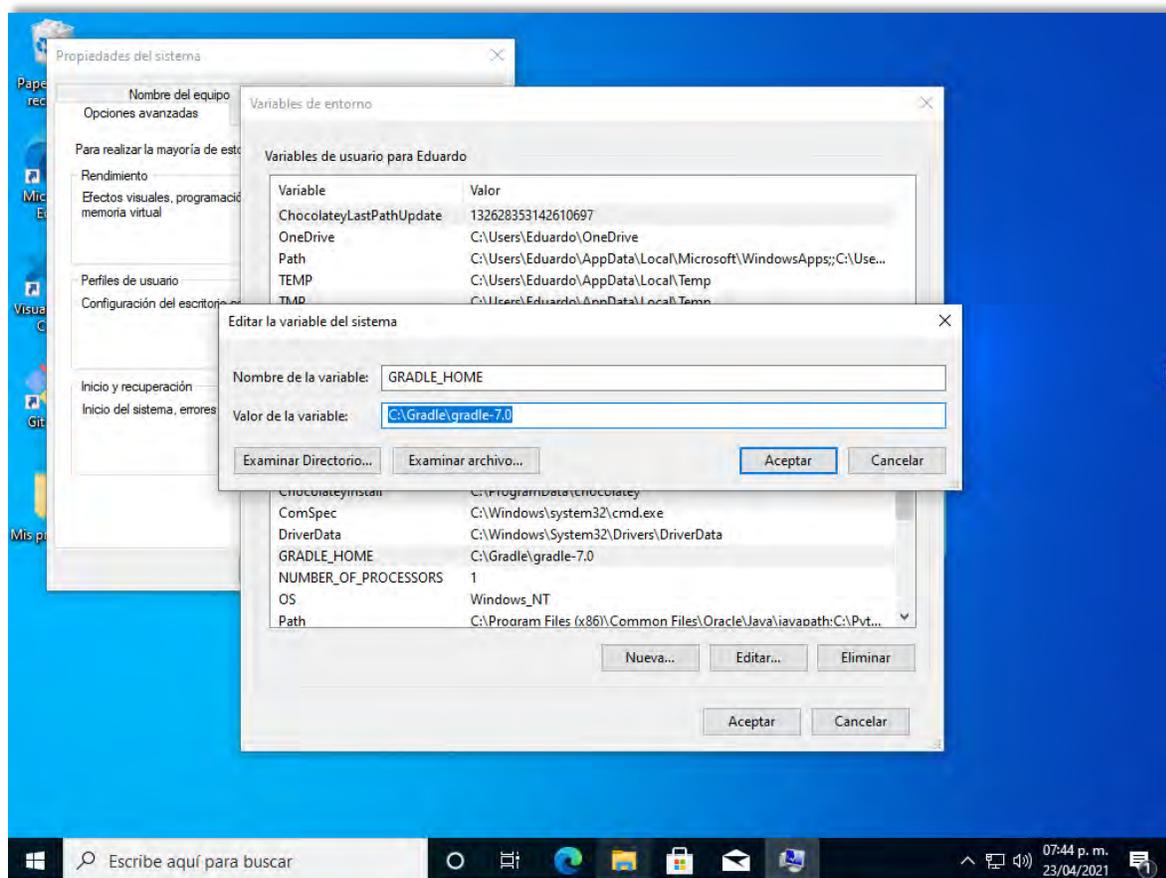


Figura 66. Agregar Gradle a las variables del sistema.

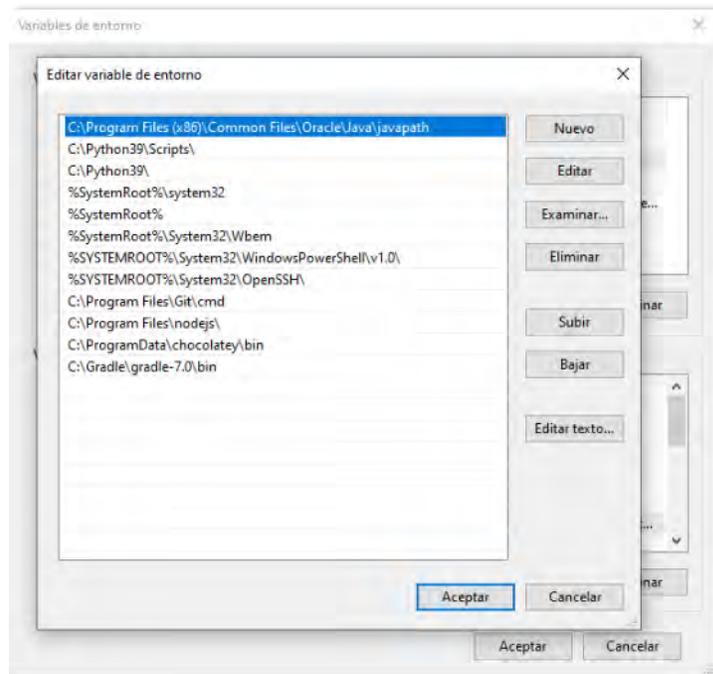


Figura 67. Agregar Gradle a la variable path del sistema.

Finalmente, para verificar la instalación ejecutamos el comando “gradle -v”, si la instalación se realizó de forma exitosa aparecerá la versión instalada en la terminal del sistema como se observa en la Figura 68.

```
Microsoft Windows [Versión 10.0.18363.1500]
(c) 2019 Microsoft Corporation. Todos los derechos reservados.

C:\Users\Eduardo>gradle -v

Welcome to Gradle 7.0!

Here are the highlights of this release:
- File system watching enabled by default
- Support for running with and building Java 16 projects
- Native support for Apple Silicon processors
- Dependency catalog feature preview

For more details see https://docs.gradle.org/7.0/release-notes.html

-----
Gradle 7.0
-----

Build time:   2021-04-09 22:27:31 UTC
Revision:    d5661e3f0e07a8caff705f1badf79fb5df8022c4

Kotlin:      1.4.31
Groovy:      3.0.7
Ant:         Apache Ant(TM) version 1.10.9 compiled on September 27 2020
JVM:         1.8.0_281 (Oracle Corporation 25.281-b09)
OS:          Windows 10 10.0 amd64

C:\Users\Eduardo>
```

Figura 68. Verificación de instalación de Gradle.

Para concluir la configuración del entorno de desarrollo de Android es necesario instalar el kit de desarrollo de java (*JDK*) descargando la aplicación desde el sitio oficial y seguir todos los pasos del instalador hasta concluir la instalación, de manera similar que en la instalación de Gradle es necesario agregar una nueva variable del sistema con la ruta hacia la carpeta en la que se encuentra la instalación de *JDK*.

Antes de generar el paquete se ejecuta el comando “`npm i -g cordova`” que instalará todas las dependencias de cordova si aun no se encuentran instaladas, concluido este paso se ejecuta el comando “`Ionic cordova build android --prod --release`”, esto generará una versión de lanzamiento del proyecto basada en la configuración del `config.xml` y que lleva por nombre “`app-release-unsigned.apk`” (ver Figura 69).

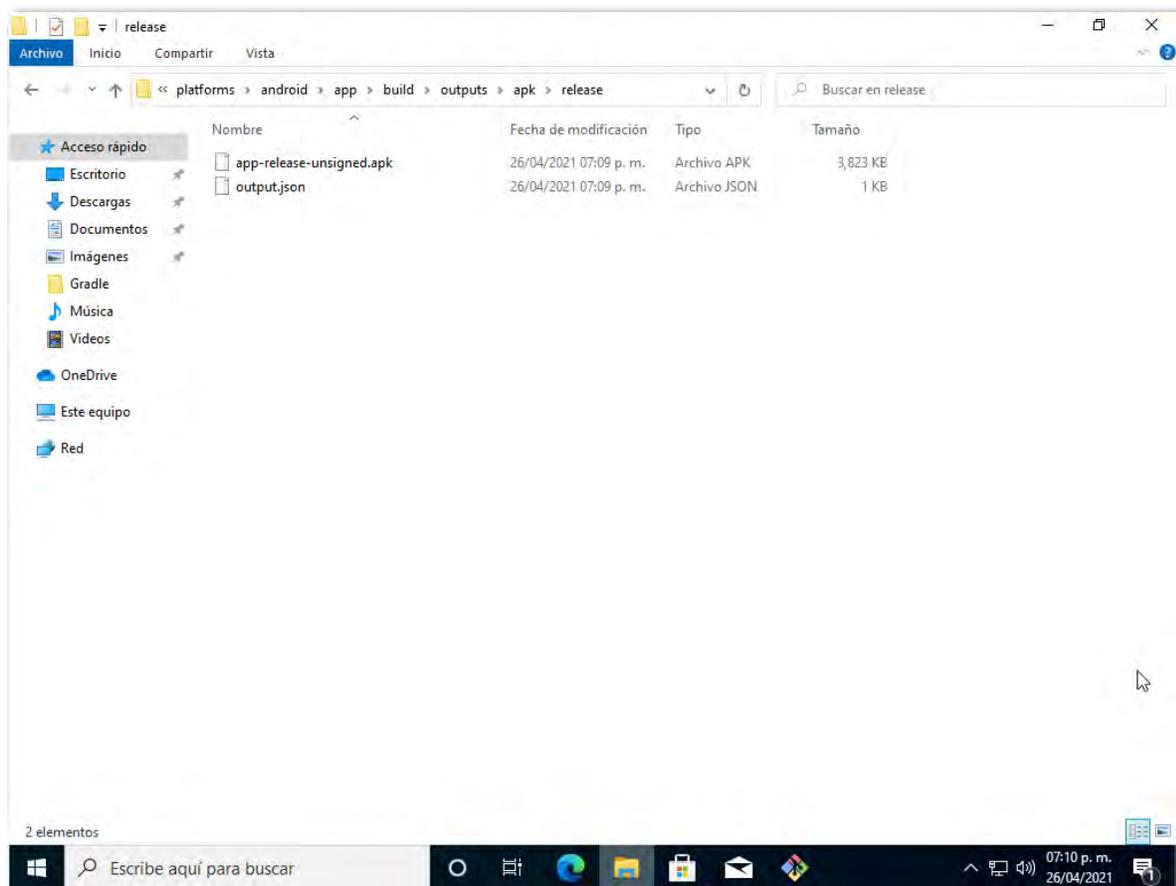


Figura 69. APK sin firmar de la aplicación móvil.

Una vez que la APK sea generada con éxito es necesario que se encuentre firmada de manera digital para poder instalarse o actualizarse en un dispositivo. Para generar una llave privada utilizamos la herramienta de línea de comandos de Android SDK ejecutando el comando que se observa en la Figura 70, esto creará un archivo llamado “my-release-key.keystore” (ver Figura 71) en el directorio que fue ejecutado, este paso se realiza una única vez y es muy importante conservar este archivo en un lugar seguro para subir actualizaciones de la aplicación en la Google Play Store.

```
$ keytool -genkey -v -keystore my-release-key.keystore -alias alias_name -  
keyalg RSA -keysize 2048 -validity 10000
```

Figura 70. comando para generar una llave privada.

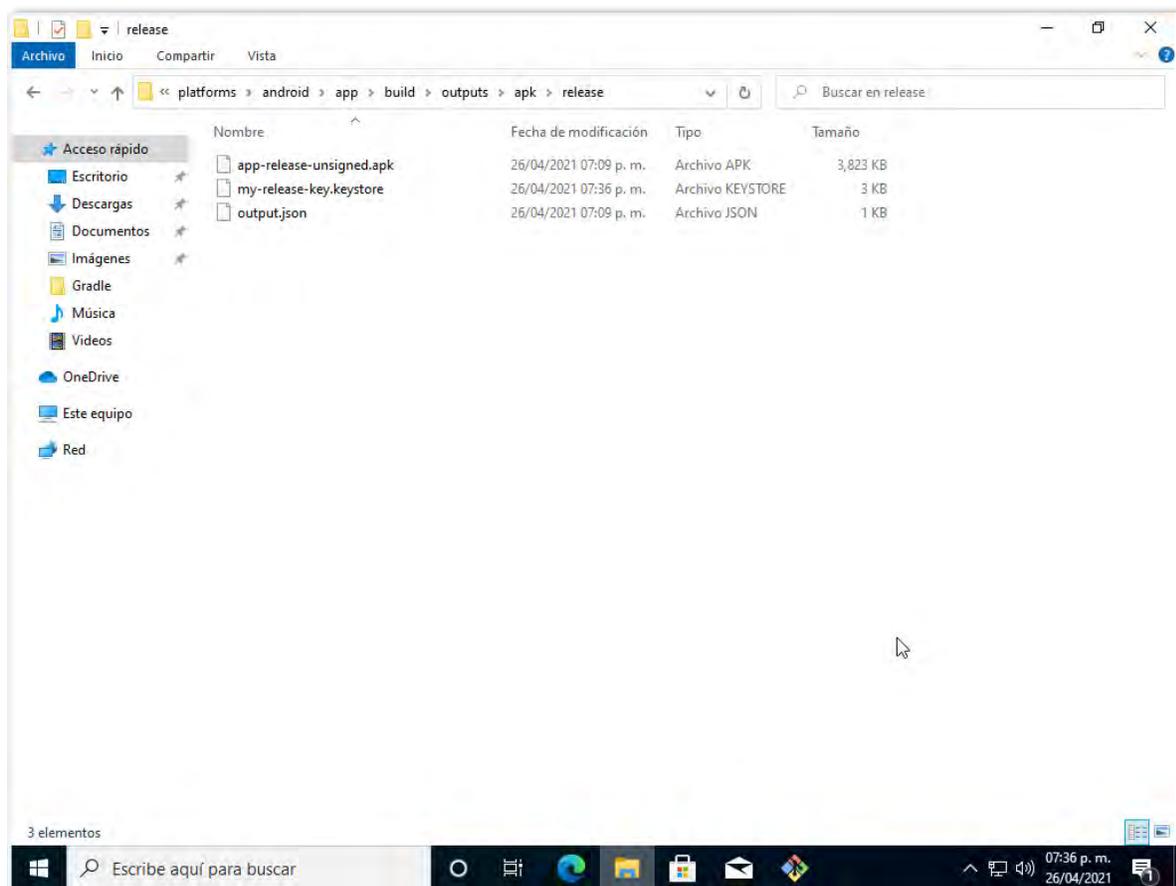


Figura 71. Llave generada para la aplicación móvil.

Una vez generada la llave se ejecuta el comando que se observa en la Figura 72, indicando el nombre de la llave y el nombre de la APK a firmar.

```
$ jarsigner -verbose -sigalg SHA1withRSA -digestalg SHA1 -keystore my-release-  
key.keystore HelloWorld-release-unsigned.apk alias_name
```

Figura 72. Comando para firmar una APK.

Finalmente, se debe ejecutar la herramienta de compresión **zipalign** para optimizar la APK generada, en la carpeta de la herramienta se copia la APK generada y se abre una nueva terminal en la que se ejecuta el comando de compresión “zipalign -v 4 app-release-unsigned.apk release.apk”. Al finalizar la ejecución de la herramienta generará un nuevo archivo APK que se encuentra listo para ser cargado a la Play Store.

Anexo 2. Infografía de la aplicación móvil.



Figura 73. Infografía de la aplicación móvil. Parte 1 de 3

Conoce nuestra nueva aplicación

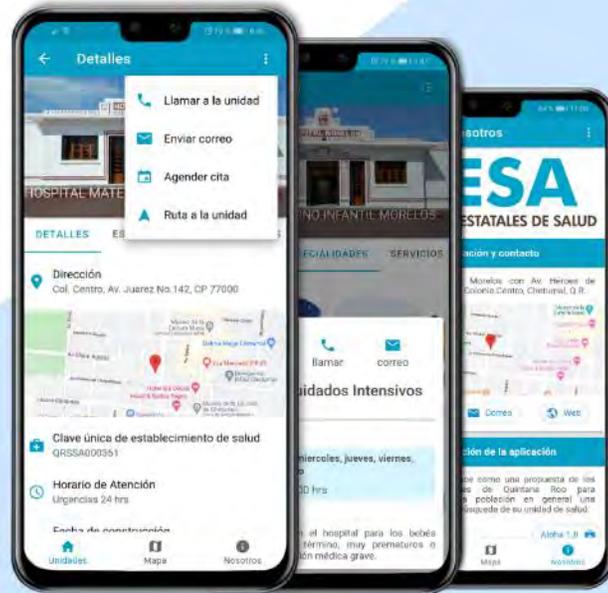
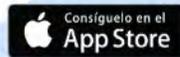
¡DESCARGÁLA YA!

iOS

<https://play.google.com/store>

Android

<https://www.apple.com/app-store/>



04 / 04

Figura 75. Infografía de la aplicación móvil. Parte 3 de 3

Anexo 3. Organigrama Servicios Estatales de Salud

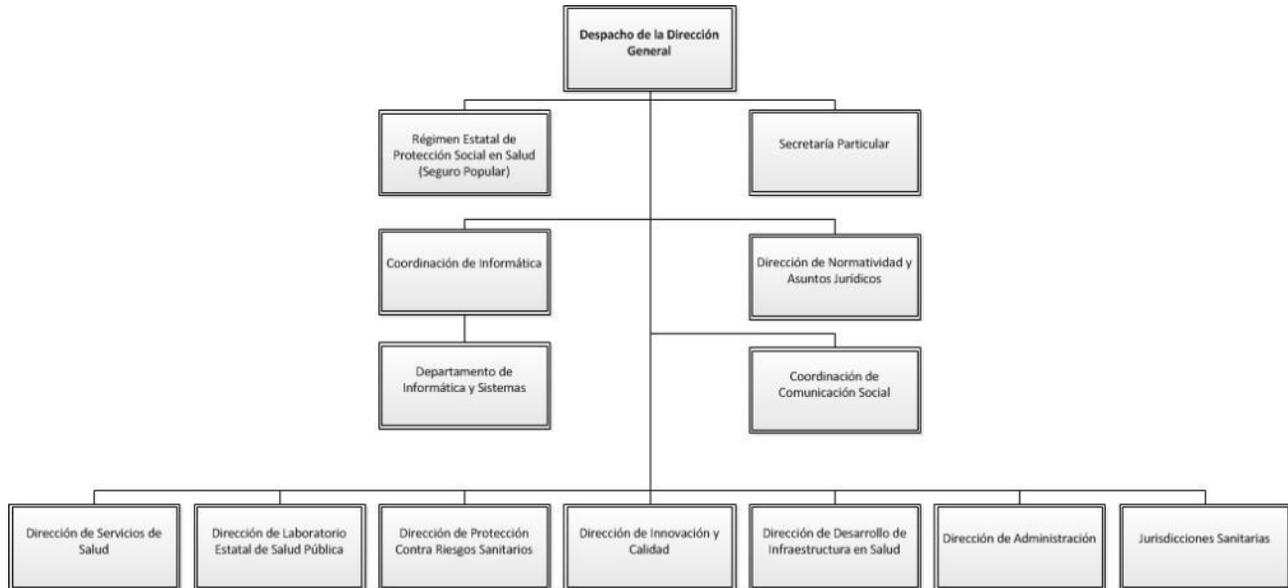


Figura 76. Organigrama SESA (SESA, 2020)

Anexo 5. Ejecución de pruebas

Tabla 7. Ejecución de la prueba detalles de una unidad de salud.

Detalles de una Unidad de Salud	
Descripción	Prueba de la funcionalidad detalles de unidad, permite verificar que la información detallada sobre la unidad de salud es consultada y mostrada de manera correcta.
Precondición	Contar con una conexión a internet.
Proceso de ejecución	
<ul style="list-style-type: none"> • Una vez cumplidas las precondiciones e iniciada la aplicación móvil el usuario debe dirigirse a la pantalla “Unidades” (🏠) o “Mapa” (📍) tocando sobre el icono correspondiente a la pantalla deseada desde la barra de navegación. • Tocar sobre la tarjeta de información de la unidad de salud deseada. 	
Resultado esperado	
<ul style="list-style-type: none"> • Se realiza la navegación a la pantalla “Detalles”. • Se muestra en pantalla la información detalla de la unidad de salud seleccionada. <ul style="list-style-type: none"> ○ Información de contacto. ○ Información general. ○ Distancia.⁴ ○ Especialidades y servicios. ○ Horarios de atención. 	
Evaluación	Prueba aprobada exitosamente.

⁴ Esta información estará disponible si el GPS se encuentra activo al ingresar a esta vista.

Tabla 8. Ejecución de la prueba filtrar unidades de salud.

Filtrar unidades de salud.	
Descripción	Prueba de la funcionalidad de filtrar para realizar una búsqueda personalizada de unidades de salud.
Precondición	Contar con una conexión a internet.
Proceso de ejecución	
<ul style="list-style-type: none"> • Una vez cumplidas las precondiciones e iniciada la aplicación móvil el usuario debe dirigirse a la pantalla de selección de filtros, accesible desde la pantalla de “Unidades” al tocar sobre el icono de lupa (Q) que se encuentra en la esquina superior derecha. • Seleccionar todos los filtros deseados y tocar sobre el icono de buscar para realizar la búsqueda. 	
Resultado esperado	
Caso 1 El usuario tiene activo el GPS al momento de realizar la búsqueda.	Se listan satisfactoriamente las unidades de salud por cercanía y se muestra únicamente las unidades que correspondan a los filtros seleccionados.
Caso 2 El usuario no tiene activo el GPS al momento de realizar la búsqueda.	Se listan satisfactoriamente las unidades de salud y se muestra únicamente las unidades que correspondan a los filtros seleccionados, pero no se ordenan por cercanía ni se incluye la distancia que hay a cada unidad de salud.
Evaluación	Prueba aprobada exitosamente.

Tabla 9. Ejecución de la prueba mapa de unidades de salud.

Mapa de unidades de salud	
Descripción	Prueba de la funcionalidad de la pantalla “Mapa” que muestra los marcadores de posición para las unidades de salud mostradas en la pantalla “Unidades”.
Precondición	Contar con una conexión a internet.
Proceso de ejecución	
<ul style="list-style-type: none"> Una vez cumplidas las precondiciones e iniciada la aplicación móvil el usuario debe dirigirse a la pantalla “Mapa” tocando sobre el icono de mapa (📍) en la barra de navegación. 	
Resultado esperado	
Caso 1 El usuario acepta los permisos de GPS solicitados por la aplicación.	Se muestra el mapa posicionado en la localidad del usuario y se marcan las unidades de salud más cercanas.
Caso 2 El usuario rechaza los permisos de GPS solicitados por la aplicación.	Se marcan las unidades de salud que se hayan buscado manualmente a través de los filtros de búsqueda y se muestra el mapa posicionado en alguna de estas unidades.
Evaluación	Prueba aprobada exitosamente.

Tabla 10. Ejecución de la prueba seguir el posicionamiento del usuario.

Seguir posicionamiento del usuario	
Descripción	Prueba para la funcionalidad de seguir el posicionamiento geográfico del dispositivo en el mapa.
Precondición	Contar con una conexión a internet.
Proceso de ejecución	
<ul style="list-style-type: none"> • Una vez cumplidas las precondiciones e iniciada la aplicación móvil el usuario debe dirigirse a la pantalla “Mapa” tocando sobre el icono de mapa (📍) en la barra de navegación. • Tocar sobre el icono “localizar” (📍) que se encuentra del lado derecho de la pantalla. 	
Resultado esperado	
Caso 1 El usuario acepta los permisos de GPS solicitados por la aplicación.	Aparece en pantalla un marcador que muestra el posicionamiento geográfico actual del dispositivo, se realiza un zoom y se centra el mapa en dicho marcador. El marcador se desplazará de acuerdo con el movimiento del usuario.
Caso 2 El usuario rechaza los permisos de GPS solicitados por la aplicación.	La aplicación mostrará una alerta indicando al usuario que es necesario habilitar el GPS para hacer uso de la funcionalidad de seguimiento de posición.
Evaluación	Prueba aprobada exitosamente.

Tabla 11. Ejecución de la prueba ruta hacia una unidad de salud.

Ruta hacia una unidad de salud	
Descripción	Prueba para la funcionalidad de trazar la ruta más cercana a una unidad de salud.
Precondición	Contar con una conexión a internet.
Proceso de ejecución	
<ul style="list-style-type: none"> • Una vez cumplidas las precondiciones e iniciada la aplicación móvil el usuario debe dirigirse a la pantalla “Unidades”, “Detalles” o “Mapa”. • Localizar y tocar sobre la opción de “Ruta” representada con el icono “navegar” (↙) que se encuentra en las pantallas anteriormente mencionadas para cada unidad de salud. 	
Resultado esperado	
Caso 1 El usuario acepta los permisos de GPS solicitados por la aplicación.	Se lanza la aplicación nativa de mapas del dispositivo móvil con una ruta trazada, que tiene que origen la ubicación del usuario y como destino la unidad de salud seleccionada.
Caso 2 El usuario rechaza los permisos de GPS solicitados por la aplicación.	Se lanza la aplicación nativa de mapas del dispositivo móvil con una ruta trazada, que tiene como destino la unidad de salud seleccionada y la aplicación nativa de mapas preguntará por el punto de partida.
Evaluación	Prueba aprobada exitosamente.

Tabla 12. Ejecución de la prueba registro de cuentas de usuario.

Registro de cuentas de usuario	
Descripción	Prueba de la funcionalidad de creación de cuentas de usuarios para la agenda de citas médicas desde la aplicación móvil.
Precondición	Contar con una conexión a internet.
Proceso de ejecución	
<ul style="list-style-type: none"> • Una vez cumplidas las precondiciones e iniciada la aplicación móvil el usuario debe dirigirse a la pantalla de “Unidades” o a la pantalla “Detalles” de una unidad de salud. • Tocar sobre el icono de calendario (📅) incluido en la tarjeta de información para cada unidad médica de la pantalla “Unidades” o sobre la opción de “agendar cita” en el menú de acciones en la pantalla “Detalles” de una unidad de salud. • En la ventana emergente tocar sobre la opción “crear una cuenta nueva” y llenar el formulario de registro en la nueva vista mostrada. • Verificar que la información obtenida de la CURP ingresada corresponda con la información personal del usuario y pulsar sobre el botón de aceptar en la alerta mostrada. 	
Resultado esperado	
<p>Caso 1</p> <p>El usuario no tiene una cuenta registrada previamente con el mismo CURP y/o correo electrónico en el sistema.</p>	<ul style="list-style-type: none"> • Se muestra una alerta para notificar la creación exitosa de la cuenta. • Se notifica vía correo electrónico la creación de la cuenta misma que debe ser verificada para concluir el proceso de registro.
<p>Caso 2</p> <p>El usuario tiene una cuenta registrada previamente con el mismo CURP y/o correo electrónico en el sistema.</p>	Se muestra una alerta notificando la existencia de una cuenta previamente registrada y se da la indicación de recuperar el acceso a su cuenta restableciendo su contraseña desde la vista de inicio de sesión.
Evaluación	Prueba aprobada exitosamente.

Tabla 13. Ejecución de la prueba restablecimiento de contraseñas.

Restablecimiento de contraseñas	
Descripción	Prueba de la funcionalidad de restablecimiento de contraseñas de cuentas de usuarios.
Precondición	Contar con una conexión a internet.
Proceso de ejecución	
<ul style="list-style-type: none"> • Una vez cumplidas las precondiciones e iniciada la aplicación móvil el usuario debe dirigirse a la pantalla de “Unidades” o a la pantalla “Detalles” de una unidad de salud. • Tocar sobre el icono de calendario () incluido en la tarjeta de información para cada unidad médica de la pantalla “Unidades” o sobre la opción de “agendar cita” en el menú de acciones en la pantalla “Detalles” de una unidad de salud. • En la ventana emergente tocar sobre la opción “recuperar contraseña” e ingresar la CURP de la cuenta previamente registrada en la nueva vista mostrada. 	
Resultado esperado	
<p>Caso 1</p> <p>El usuario tiene una cuenta registrada con la CURP ingresada en el sistema.</p>	<ul style="list-style-type: none"> • Se muestra una alerta para notificar que fue enviado de manera exitosa un enlace de restablecimiento de contraseña a la dirección de correo electrónico asociada a la CURP ingresada. • El usuario deberá ingresar a su correo electrónico, hacer clic en el enlace recibido y llenar el formulario de restablecimiento ingresando una nueva contraseña.
<p>Caso 2</p> <p>El usuario no tiene una cuenta registrada con la CURP ingresada en el sistema.</p>	Se muestra una alerta notificando la ausencia de una cuenta con la CURP ingresada en el sistema.
Evaluación	Prueba aprobada exitosamente.

Tabla 14. Ejecución de la prueba agendar una cita médica.

Agendar una cita médica en una unidad de salud	
Descripción	Prueba de la funcionalidad para agendar citas médicas desde la aplicación móvil.
Precondición	Contar con una conexión a internet.
	Contar con una cuenta verificada.
Proceso de ejecución	
<ul style="list-style-type: none"> • Una vez cumplidas las precondiciones e iniciada la aplicación móvil el usuario debe dirigirse a la pantalla de “Unidades” o a la pantalla “Detalles” de una unidad de salud. • Tocar sobre el icono de calendario (📅) incluido en la tarjeta de información para cada unidad médica de la pantalla “Unidades” o sobre la opción de “agendar cita” en el menú de acciones de la pantalla “Detalles” de una unidad de salud. • Iniciar sesión en la ventana emergente que es lanzada. • Seleccionar la fecha deseada en el calendario mostrado. 	
Resultado esperado	
<p>Caso 1</p> <p>La unidad cuenta citas disponibles para el día seleccionado.</p>	<ul style="list-style-type: none"> • Se muestra el listado de horarios disponibles en el día, el usuario selecciona una hora y confirma la creación de la cita médica. • Se notifica vía correo electrónico la creación exitosa de la cita médica y se envían todos los detalles referentes a la misma. • Se muestra una alerta para notificar la creación exitosa de la cita médica en la aplicación móvil.
<p>Caso 2</p> <p>El usuario ya cuenta con una cita médica agendada.</p>	Se muestra una alerta indicando al usuario el problema y se le brinda la opción de cancelar su cita actual para agendar una nueva o reenviar la información detalla de la cita al correo electrónico.
<p>Caso 3</p> <p>La unidad no cuenta con citas disponibles en el día seleccionado.</p>	Se muestra una alerta indicando al usuario la usencia de citas en el día seleccionado y se hace la sugerencia de seleccionar un día diferente.
<p>Caso 4</p> <p>La unidad no cuenta con el servicio de consulta externa.</p>	Se muestra una alerta indicando al usuario la usencia del servicio en la unidad y se hace la sugerencia de seleccionar una unidad diferente.
Evaluación	Prueba aprobada exitosamente.