

UNIVERSIDAD TÉCNICA DEL NORTE



Facultad de Ingeniería en Ciencias Aplicadas
Carrera de Ingeniería en Sistemas Computacionales

“ESTUDIO COMPARATIVO DE LOS FRAMEWORKS IONIC Y REACT NATIVE” APLICACIÓN MÓVIL DE PEDIDOS A DOMICILIO BASADA EN LA NORMA ISO 9126

Trabajo de grado previo a la obtención del título de ingeniero en Sistemas
Computacionales

Autor:

Robinson Vicente Ruano Valenzuela

Directora:

Ing. Daisy Elizabeth Imbaquingo Esparza, MSC.

Ibarra – Ecuador

2018



UNIVERSIDAD TÉCNICA DEL NORTE

BIBLIOTECA UNIVERSITARIA

AUTORIZACIÓN DE USO Y PUBLICACIÓN

A FAVOR DE LA UNIVERSIDAD TÉCNICA DEL NORTE

1.- IDENTIFICACIÓN DE LA OBRA

En cumplimiento del Art. 144 de la Ley de Educación Superior, hago la entrega del presente trabajo a la Universidad Técnica del Norte para que sea publicado en el Repositorio Digital Institucional, para lo cual pongo a disposición la siguiente información:

DATOS DE CONTACTO	
CÉDULA DE IDENTIDAD:	040170969-6
APELLIDOS Y NOMBRES:	RUANO VALENZUELA ROBINSON VICENTE
DIRECCIÓN:	IBARRA-OLMEDO 333 y OVIEDO
EMAIL:	rvruanov@utn.edu.ec
TELÉFONO MOVIL:	0999773870
DATOS DE LA OBRA	
TÍTULO:	“ESTUDIO COMPARATIVO DE LOS FRAMEWORKS IONIC Y REACT NATIVE” APLICACIÓN MÓVIL DE PEDIDOS A DOMICILIO BASADA EN LA NORMA ISO 9126
AUTOR:	ROBINSON VICENTE RUANO VALENZUELA
FECHA:	NOVIEMBRE DEL 2018
PROGRAMA:	<input type="checkbox"/> PREGRADO <input type="checkbox"/> POSTGRADO
TÍTULO POR EL QUE OPTA:	INGENIERIA EN SISTEMAS COMPUTACIONES
DIRECTORA:	ING. DAISY IMBAQUINGO, MSC.

2.- CONSTANCIAS

El autor manifiesta que la obra objeto de la presente autorización es original y se la desarrolló, sin violar derechos de autor de terceros, por lo tanto, la obra es original y que es el titular de los derechos patrimoniales, por lo que asume la responsabilidad sobre el contenido de la misma y saldrá en defensa de la Universidad en caso de reclamación por parte de terceros. Ibarra, a los 27 días del mes de noviembre de 2018

El autor:




(Firma).....

Nombre: Robinson Vicente Ruano Valenzuela

Cedula: 0401709696

CERTIFICACIÓN

El señor Robinson Vicente Ruano Valenzuela con cédula de identidad 040170969-6, ha trabajado en el desarrollo del proyecto de grado **“ESTUDIO COMPARATIVO DE LOS FRAMEWORKS IONIC Y REACT NATIVE” APLICACIÓN MÓVIL DE PEDIDOS A DOMICILIO BASADA EN LA NORMA ISO 9126**, previo a la obtención de Título de Ingeniero en Sistemas Computacionales, realizándola con interés profesional y responsabilidad, que certifico en honor a la verdad.

A handwritten signature in purple ink, appearing to read 'Daisy Imbaquingo', is positioned above a horizontal line.

Ing. Daisy Imbaquingo, Msc.

DIRECTORA DE TESIS

CERTIFICACIÓN



Ibarra, 20 de agosto del 2018

Señores

UNIVERSIDAD TECNICA DEL NORTE

Presente

De mis consideraciones. -

Siendo auspiciantes del proyecto de trabajo de grado del señor Robinson Vicente Ruano Valenzuela con Nro. de cédula 040170969-6, quien desarrollo su trabajo con el tema **“ESTUDIO COMPARATIVO DE LOS FRAMEWORKS IONIC Y REACT NATIVE” APLICACIÓN MÓVIL DE PEDIDOS A DOMICILIO BASADA EN LA NORMA ISO 9126**, me es grato informar que se han superado con satisfacción las pruebas técnicas y la revisión de cumplimiento de los requerimientos funcionales, por lo que se recibe el proyecto como culminado y realizado por parte del señor Robinson Vicente Ruano Valenzuela. Una vez que hemos recibido la capacitación y la documentación respectiva, nos comprometemos a continuar utilizando el mencionado aplicativo en beneficio de nuestra empresa.

El señor Robinson Vicente Ruano Valenzuela puede hacer uso de este documento para los fines pertinentes en la Universidad Técnica del Norte.

Atentamente,

Sr. José Orbe Guerrón
Gerente La Lico Distribuidora

AGRADECIMIENTO

A mis padres por estar presentes incondicionalmente y brindarme todo el apoyo en el transcurso de mi formación como profesional.

Un agradecimiento muy especial a mi directora de trabajo de grado, la MSc. Daisy Imbaquingo que ha orientado, apoyado y guiado en este trabajo con mucha dedicación y una entrega que ha cumplido todas mis expectativas.

A la Universidad Técnica del Norte por ofrecerme un entorno adecuado donde pude instruirme día a día.

Muchas gracias a todos mis compañeros con quienes compartí momentos inolvidables en las aulas de clases y fuera de ellas.

Robinson Vicente Ruano Valenzuela

Resumen

En el presente trabajo de grado es el “Estudio Comparativo de los Frameworks Ionic y React Native”, para desarrollar una aplicación móvil de pedidos a domicilio para la distribuidora “La Lico”. El proyecto consta de los siguientes capítulos:

En la Introducción, se declara el objetivo general y específicos; se detallan los problemas que han generado la falta de automatización del proceso de pedidos a domicilio y se plantea una solución.

En el Capítulo 1, se describe toda la teoría en la que se basa el estudio comparativo y la información de las herramientas que se utilizó en el proyecto.

En el Capítulo 2, se realiza el estudio comparativo basado en la norma ISO 9126 utilizando las métricas de calidad y se determina el software ganador que será utilizado para la ejecución del proyecto.

En el Capítulo 3, se implementa y se documenta el desarrollo de la aplicación utilizando la metodología XP.

Al final del documento se presentan las conclusiones y recomendaciones que se basan en el estudio comparativo y la implementación de la aplicación.

PALABRAS CLAVES: Ionic, React Native, ISO 9126, aplicación móvil.

ABSTRACT

In the present degree work is the "Comparative study of Ionic and React Native Frameworks", to develop a mobile application of orders for the "La Lico" distributor store. The project consists of the following chapters:

In the Introduction, the general and specific objective is declared; the problems that have caused the lack of automation of the ordering process at home are detailed and a solution is proposed.

In Chapter 1, it is described the whole theory on which the comparative study and the information of the tools used in the project are based.

In Chapter 2, the comparative study is based on the ISO 9126 standard, it is carried out using the quality metrics and the winning software that will be used for the execution of the project is determined.

In Chapter 3, the development of the application using the XP methodology is implemented and documented.

At the end of the document, the conclusions and recommendations are based on the comparative study and the implementation of the application are presented.

KEY WORDS: Ionic, React Native, ISO 9126, mobile app.

INDICE DE CONTENIDOS

AUTORIZACIÓN DE USO Y PUBLICACIÓN.....	II
CERTIFICACIÓN	IV
CERTIFICACIÓN	V
AGRADECIMIENTO.....	VI
Resumen.....	VII
ABSTRACT	VIII
INDICE DE CONTENIDOS.....	IX
ÍNDICE DE TABLAS.....	XII
ÍNDICE DE FIGURAS	XV
Introducción.....	XVI
CAPÍTULO I	1
Marco Teórico	1
1.1. Dispositivos Móviles	1
1.1.1. Teléfonos Móviles.....	1
1.1.2. Tableta	2
1.2. Sistemas Operativos Móviles.....	2
1.2.1 Android.....	3
1.2.2. iOS.....	4
1.2.3. Windows Phone.....	5
1.2.4. Blackberry OS.....	5
1.3. Aplicaciones Móviles	5
1.3.1. Tipos De Aplicaciones	6
1.4. Framework	7
1.4.1. Ventajas De Utilizar Un Framework	7
1.5. Framework React Native	8
1.5.1. Componentes	9
1.5.2. JSX.....	9

1.5.3. Data Flow de React Native	9
1.5.4. Arquitectura Flux en React Native	10
1.5.5. Instalación del Framework React Native.....	11
1.6. Framework Ionic.....	11
1.6.1. Apache Cordova.....	12
1.6.2. Angular 4	12
1.6.3. Arquitectura de Aplicaciones Creadas con Ionic.....	13
1.6.4. Typescript.....	13
1.6.5. Instalación del Framework Ionic.....	14
1.7. Metodología XP	14
1.7.1. Valores	14
1.7.2. Características Fundamentales	15
1.7.3. Roles.....	16
1.8. Norma ISO 9126.....	16
1.8.1. Portabilidad	17
1.8.2. Funcionalidad	18
1.8.3. Fiabilidad	18
1.8.4. Usabilidad.....	19
1.8.5. Eficiencia	19
1.8.6. Mantenibilidad	20
CAPÍTULO II	21
Estudio Comparativo	21
2.1. Escala de Valoración.....	21
2.2. Parámetros de Evaluación.....	22
2.3. Análisis Comparativo	23
2.3.1. Portabilidad	23
2.3.2. Funcionalidad	24
2.3.3. Fiabilidad	25

2.3.4. Usabilidad.....	26
2.3.5. Eficiencia	27
2.3.6. Mantenibilidad	27
2.4. Determinación del Software Ganador	27
CAPÍTULO III	29
Desarrollo Del Sistema	29
3.1. Planificación del Proyecto.....	29
3.1.1. Roles de Usuarios	29
3.1.2. Tipos de Usuarios.....	29
3.1.3. Historias de Usuarios.....	29
3.1.4. Plan de Entregas	32
3.1.5. Módulos Del Sistema.....	33
3.1.6. Planificación: Iteración I	34
3.1.7. Planificación: Iteración II.....	41
3.2. Diseño de la Aplicación	50
3.2.1. Arquitectura de la Aplicación	50
3.2.2. Diagrama de Base De Datos	52
3.2.3. Funcionamiento de la Aplicación	52
3.2.4. Diseño de la Aplicación.....	53
3.3. Pruebas.....	54
3.3.1. Prueba Iteración I	54
3.3.2. Prueba Iteración II	60
CONCLUSIONES.....	67
RECOMENDACIONES	68
REFERENCIAS	69
ANEXOS	73
Anexo A: Capturas de pantalla de la aplicación desarrollada con React Native.....	73

ÍNDICE DE TABLAS

TABLA 1.1 Sub-características de portabilidad	17
TABLA 1.2 Sub-características de funcionalidad.....	18
TABLA 1.3 Sub-características de fiabilidad.....	18
TABLA 1.4 Sub-características de usabilidad.....	19
TABLA 1.5 Sub-características de eficiencia.....	19
TABLA 1.6 Sub-características de mantenibilidad.....	20
TABLA 2.7 Ejemplo de la escala de Likert	22
TABLA 2.8 Ejemplo de tabla de evaluación	22
TABLA 2.9 Análisis de Portabilidad.....	23
TABLA 2.10 Análisis de Funcionalidad	24
TABLA 2.11 Análisis de Fiabilidad	25
TABLA 2.12 Análisis de Usabilidad	26
TABLA 2.13 Análisis de Eficiencia	27
TABLA 2.14 Análisis del Software Ganador.....	28
TABLA 3.15 Descripción de roles de usuarios	29
TABLA 3.16 Historia de usuario 01	30
TABLA 3.17 Historia de usuario 02	30
TABLA 3.18 Historia de usuario 03	30
TABLA 3.19 Historia de usuario 04	31
TABLA 3.20 Historia de usuario 05	31
TABLA 3.21 Historia de usuario 06	31
TABLA 3.22 Historia de usuario 07	32
TABLA 3.23 Equivalencias de tiempo de desarrollo.....	32
TABLA 3.24 Plan de entregas.....	33
TABLA 3.25 Cronograma de Iteración I.....	34
TABLA 3.26 Tareas de historia 1	34
TABLA 3.27 Tareas de historia 1 Tarea 1 configuración del entorno de Programación.....	35

TABLA 3.28 Tareas de historia 1 Tarea 2 configuración del servidor	35
TABLA 3.29 Tareas de historia 1 Tarea 3 diseño de interfaz gráfica, aplicación de plantilla	36
TABLA 3.30 Tareas de historia 1 Tarea 4 implementación del módulo de login	36
TABLA 3.31 Tareas de historia 2	37
TABLA 3.32 Tareas de historia 2 Tarea 1 diseño de interfaz gráfica del portal	37
TABLA 3.33 Tareas de historia 2 Tarea 2 diseño de botones para cada opción	38
TABLA 3.34 Tareas de historia 2 Tarea 3 Implementación de CRUD para productos, sucursales y ciudades.....	38
TABLA 3.35 Tareas de historia 2 Tarea 4 Implementar página de estadísticas para el administrador	39
TABLA 3.36 Tareas de historia 3	39
TABLA 3.37 Tareas de historia 3 Tarea 1 Diseño del módulo de Visualización de Pedidos	40
TABLA 3.38 Tareas de historia 3 Tarea 2 Diseño de botones para cada opción.....	40
TABLA 3.39 Tareas de historia 3 Tarea 3 Implementación de funciones principales.....	41
TABLA 3.40 Cronograma iteración II.....	41
TABLA 3.41 Tareas de historia 4	42
TABLA 3.42 Tareas de historia 4 Tarea 1 Configuración del entorno de programación	42
TABLA 3.43 Tareas de historia 4 Tarea 2 Diseño de interfaz gráfica del login	43
TABLA 3.44 Tareas de historia 4 Tarea 3 Implementación del código de verificación de acceso	43
TABLA 3.45 Tareas de historia 5	44
TABLA 3.46 Tareas de historia 5 Tarea 1 Diseño de interfaz gráfica del menú.....	44
TABLA 3.47 Tareas de historia 5 Tarea 2 Implementación de animación y transiciones.....	45
TABLA 3.48 Tareas de historia 5 Tarea 3 Implementación de funcionalidad del menú	45
TABLA 3.49 Tareas de historia 6	46
TABLA 3.50 Tareas de historia 6 Tarea 1 Diseño del módulo de Visualización de Productos	46
TABLA 3.51 Tareas de historia 6 Tarea 2 Diseño de botones para cada opción.....	47
TABLA 3.52 Tareas de historia 6 Tarea 3 Creación de Web Service Rest	47

TABLA 3.53 Tareas de historia 6 Tarea 4 Implementación de funciones principales.....	48
TABLA 3.54 Tareas de historia 7	48
TABLA 3.55 Tareas de historia 7 Tarea 1 Diseño de botones para cada opción	49
TABLA 3.56 Tareas de historia 7 Tarea 1 Diseño de botones para cada opción.....	49
TABLA 3.57 Tareas de historia 7 Tarea 1 Implementación de funciones principales.....	50
TABLA 3.58 Prueba de ingresar a la aplicación web.....	54
TABLA 3.59 Prueba de visualización de datos en el sistema	55
TABLA 3.60 Prueba de gestión de datos en el sistema.....	56
TABLA 3.61 Prueba de estadísticas.....	57
TABLA 3.62 Prueba de visualización de pedidos	58
TABLA 3.63 Prueba de visualización del detalle de cada pedido	59
TABLA 3.64 Prueba de Acceso a la Aplicación del Cliente	60
TABLA 3.65 Prueba de registro en la aplicación	61
TABLA 3.66 Prueba de visualización del menú de opciones.....	62
TABLA 3.67 Prueba de visualización de productos y promociones	63
TABLA 3.68 Prueba Agregar al carrito de compras.....	64
TABLA 3.69 Prueba realizar pedidos	65

ÍNDICE DE FIGURAS

Fig.1 Arquitectura de aplicaciones generadas por el framework React Native	11
Fig 2. Arquitectura de aplicaciones generadas por el framework Ionic.....	13
Fig 3. Características de la ISO 9126	17
Fig 4. Arquitectura de la Aplicación.....	51
Fig 5. Diagrama de la base de datos	52
Fig 6. Diagrama del Módulo Administrador	53
Fig 7. Diagrama del Módulo Cliente.....	53
Fig 8. Login del portal web.....	54
Fig 9. Gestor de productos	55
Fig 10. Ingreso de productos	56
Fig 11. Módulo de estadísticas	57
Fig 12. Módulo de historial de pedidos	58
Fig 13. Detalle del pedido	59
Fig 14. Login de la aplicación web	60
Fig 15. Registro de clientes	61
Fig 16. Menú de la aplicación móvil	62
Fig 17. Visualización de productos	63
Fig 18. Agregar al carrito	64
Fig 19. Detalle del pedido	65

Introducción

Antecedentes

Con el paso del tiempo se han desarrollado varias herramientas nuevas para la construcción de aplicaciones móviles, las cuales ayudan a generar aplicaciones que sirven para varias plataformas sin perder el rendimiento de una aplicación nativa, por el crecimiento de nuevas herramientas muchos programadores no conocen las ventajas que ofrecen estos frameworks.

Las aplicaciones desarrolladas en Ionic se crean principalmente a través de la utilidad de línea de comandos Ionic (la "CLI") y usan Cordova¹ para compilar e implementar como una aplicación nativa para plataformas iOS y Android.

React crea una aplicación móvil real que no se distingue de una aplicación creada con Objective-C² o Java. React Native utiliza los mismos bloques de interfaz de usuario que las aplicaciones normales de iOS y Android. Simplemente se agrupan los bloques usando JavaScript³ y React.

La distribuidora "La Lico" lleva el proceso de recepción de pedidos a domicilio de una forma manual, utilizando redes sociales y no se tiene una automatización del proceso, por lo que están expuestos a encontrarse con varios errores y pérdidas de información a lo largo de la transacción. Estos problemas ocurren por falta de un sistema que permita automatizar la recepción de pedidos a domicilio.

Prospectiva

Mediante la aplicación del trabajo de titulación con la implementación del sistema de pedidos a domicilio, permitirá a los clientes de Ibarra tener accesibilidad a los productos y poder realizar pedidos desde un smartphone automatizando así, el proceso de recepción de pedidos.

A través de la aplicación los clientes podrán observar el estado de su pedido.

Por medio de geocalización los repartidores podrán ver la dirección exacta de donde deben hacer sus entregas.

¹ **Cordova:** Framework de desarrollo de aplicaciones móviles.

² **Objective-C:** Lenguaje de programación desarrollado por Appel Inc.

³ **JavaScript:** Lenguaje de programación.

Los administradores participantes podrán observar estadísticas de los pedidos que se han realizado en un tiempo determinado.

Planteamiento del Problema

¿Con el estudio de los dos frameworks de desarrollo móvil planteados se podrá determinar cuál de las dos herramientas es la mejor opción para realizar una aplicación móvil?

Objetivos

Objetivo General

Comparar los frameworks Ionic y React Native y demostrar mediante una aplicación móvil de pedidos a domicilio basado en la norma ISO 9126.

Objetivos Específicos

- Investigar las herramientas Ionic y React Native.
- Determinar métricas de comparación.
- Investigar cómo se aplica la metodología XP.
- Seleccionar la herramienta a ser utilizada en la implementación.
- Emplear la herramienta seleccionada para el desarrollo de la aplicación móvil.

Alcance

Se realizará una comparación entre los frameworks Ionic y React Native basándose en los parámetros establecidos por la norma ISO 9126 para determinar la calidad de cada uno de los frameworks, con el fin de seleccionar una herramienta que será usada para el desarrollo de una aplicación móvil.

La aplicación móvil será capaz de listar los productos que estén disponibles para la venta para que el usuario pueda seleccionar y agregarlos al carrito. El usuario podrá realizar pedidos donde se detallará la información sobre la factura generada con la ubicación actual del cliente.

Módulo De Cliente

En este módulo los usuarios registrarán datos básicos y tendrá las opciones de realizar pedidos a domicilios, ver sus pedidos pendientes.

Módulo de Administración

En este módulo los administradores podrán revisar estadísticas de las transacciones que se han realizado en un periodo de tiempo, visualizar los pedidos y publicar productos, este módulo será un complemento de la aplicación por este motivo será web.

Justificación

Justificación Tecnológica

La llegada de las nuevas tecnologías móviles y el mejoramiento del rendimiento de estos dispositivos, han provocado que dichos dispositivos se conviertan en una extensión más del ser humano debido a que ofrecen muchas opciones que hacen que la vida diaria sea más fácil.

Justificación Teórica

Con la llegada de los sistemas operativos móviles como: iOS y Android se facilita el consumo de recursos ofrecidos en la web, al igual de la obtención de servicios que ofrecen varias empresas. Para esto se necesita las mejores herramientas de desarrollo para obtener una aplicación de calidad que ayude a los usuarios con una experiencia fluida y completa al momento de solicitar un servicio.

Justificación Metodológica

Se utilizará la metodología XP eXtreme Programming para el desarrollo de la aplicación debido a que esta metodología es usada en proyectos pequeños y se basa en el desarrollo continuo del sistema.

CAPÍTULO I

Marco Teórico

1.1. Dispositivos Móviles

Los dispositivos móviles se definen como aparatos eléctricos de tamaño reducido que tienen la capacidad de procesar información, algunos de estos dispositivos cuentan con conexión a la red. (Domínguez, 2014). Estos dispositivos han sido diseñados para realizar una función específica, además, de realizar pequeñas funciones adicionales básicas. Con la siguiente definición se da a entender como dispositivo móvil a cualquier dispositivo que procesa información y es fácil de transportar. En esta categoría se encuentran desde los reproductores de música, teléfonos móviles, gafas de Google, hasta computadoras portátiles.

Al pasar de los años desde su creación estos dispositivos han tenido una gran acogida por los usuarios. En el año 2013, por primera vez en la historia, el número de dispositivos móviles conectados, en su mayor parte teléfonos móviles, ha superado el número de habitantes del planeta. (UNESCO, 2013).

Desde los enormes terminales móviles a los teléfonos inteligentes, los teléfonos han recorrido un largo camino, en relativamente no demasiado tiempo. Durante ese recorrido, las tecnologías han ido mejorando para ofrecer al usuario una amplia gama de prestaciones. Así, hemos pasado de dispositivos con grandes dimensiones, a dispositivos con precios al alcance de los usuarios y cada vez mejores prestaciones. (Domínguez, 2014, pág. 12).

Esta investigación se centrará en los dispositivos que dan soporte los frameworks Ionic y React Native como son: teléfonos móviles y tables por el motivo de que estos aparatos electrónicos son muy comunes en la actualidad y permiten instalar aplicaciones de diversos tipos.

1.1.1. Teléfonos Móviles

“Los Smartphones, teléfonos avanzados o inteligentes, son dispositivos móviles diseñados inicialmente para mantener comunicaciones biunívocas de mensajes visuales, sonoros o textuales” (Vázquez-Cano & Sevillano, 2015,). En sus principios fueron diseñados para realizar llamadas sin necesidad de estar conectado a otro terminal por medio de un cable, con el paso del tiempo se implementaron nuevas funciones como mensajería instantánea, juegos, linterna, cámara entre otras.

Una característica principal de este tipo de dispositivos es permitir la instalación de programas o aplicaciones que realizan el procesamiento de información debido a que poseen

un sistema operativo que facilita la interacción con el equipo, además, de contar con gran cantidad de sensores.

Por las prestaciones que nos brindan estos tipos de dispositivos y su fácil acceso ha provocado que los programadores enfoquen esfuerzos para desarrollar aplicaciones para estos dispositivos, dando como resultado que estén disponibles 2888368 aplicaciones en la Play Store (Julio del 2018).

1.1.2. Tableta

Una tableta es un dispositivo informático cuyo canal de comunicación con el usuario es una pantalla táctil. Esta superficie plana es el origen de su denominación. (Vázquez-Cano & Sevillano, 2015, p.67). Las tabletas comparten casi todas las características de un teléfono inteligente con la diferencia de que sus pantallas son de un tamaño superior permitiendo una mejor interacción para el usuario.

1.2. Sistemas Operativos Móviles

Un sistema operativo es el programa que primero se ejecuta en un computador o dispositivo. Wolf, Ruiz, Bergero & Meza (2015) afirman: “El sistema operativo es el único programa que interactúa directamente con el hardware de la computadora.” (p.18). Con la finalidad de administrar los recursos del dispositivo y facilitar al usuario la interacción con el dispositivo y poder procesar la información según lo requiera.

“Se trata de sistemas dispositivo. operativos diseñados para computadores de mano (como por ejemplo tabletas y relojes inteligentes, entre otros). Están optimizados para mejorar el uso de la energía y atender a las limitaciones del dispositivo” (García, Gómez, Rubén, & Molina, 2017, pág. 140). Por este motivo los sistemas operativos móviles tienen una interfaz de usuario más simple debido a que están más orientados a la conectividad inalámbrica y a la reproducción de contenido multimedia, sin dejar de lado la parte productiva permitiendo también realizar actividades como procesamiento de datos, visualización de documentos y cálculos numéricos.

Cada diseñador de smartphones decide el sistema operativo que instalará en su producto por este motivo las empresas han desarrollado varios sistemas operativos de este tipo para satisfacer las necesidades de los usuarios. Los más usados en la actualidad son Android, iOS y Windows Phone y otros no tan usados como BlackBerry OS, Firefox OS entre otros que últimamente ya están desapareciendo del mercado.

1.2.1 Android

“Android es un sistema operativo móvil que tuvo gran acogida, comenzó como un pequeño proyecto hasta que fue comprada por Google en 2005 desde ese momento fue creciendo y madurando en función del tiempo” (Domínguez, 2014, pág. 19). Al principio fue diseñado para teléfonos móviles con pantallas táctiles, pero con el paso del tiempo se ha instalado en varios dispositivos como GPS, televisores, miniordenadores incluso en lavadoras y microondas.

“El sistema permite ejecutar aplicaciones escritas en Java (por medio virtual propia) y, dado que Java⁴ es un lenguaje muy conocido, atrae a muchos programadores” (García, Gómez, Rubén, & Molina, 2017).

“El núcleo de Android está basado en el de Linux para el manejo de memoria, procesos y hardware. (Se trata de una rama, de manera que las mejoras introducidas no se incorporan en el desarrollo del núcleo de GNU/Linux)” (Dpto de Ciencia de la Computación e Inteligencia Artificial, 2018). Montando así una arquitectura sobre dicho núcleo. La arquitectura de Android está formada por cuatro capas que son las siguientes:

- **Núcleo Linux:** El núcleo de Android está basado en la versión 2.6 de Linux y permite realizar los procesos básicos de un sistema operativo tales como manejo de memoria, la pila de protocolos y el soporte de drivers. (Domínguez, 2014). Es decir es capa de abstracción que comunica el hardware con el software.
- **Runtime de Android y Librerías Nativas:** Esta capa se forma por el runtime de Android que se basa en la máquina virtual de Java llamada Dalvik que es más ligera, para manejar mejor los problemas con las limitaciones de memoria y procesos. También contiene las librerías nativas que incluyen librerías de C/C++⁵ que están compiladas en el código nativo del procesador. (Sanz & López, 2014)
- **Entorno de Aplicación:** Esta capa fue diseñada para facilitar que los componentes sean reutilizados. Siendo así un conjunto de aplicaciones que ofrecen acceso al hardware y a la información del dispositivo. (Domínguez, 2014)
- **Aplicaciones:** Este nivel está formado por el conjunto de aplicaciones instaladas en una máquina Android. Todas las aplicaciones han de ser ejecutadas en la máquina virtual Dalvik para garantizar la seguridad del sistema. (Gironés, 2014). En esta capa se encuentran todas las funciones que tiene el teléfono como lectores de texto,

⁴ **Java:** Lenguaje de programación

⁵ **C/C++:** Lenguajes de programación

navegadores, calculadoras, etc. Por lo general estas aplicaciones están sobre el lenguaje Java, pero existen opciones que se describirán a lo largo del documento.

1.2.2. iOS

iOS en sus principios fue llamado iPhone OS cuando salió el 9 de enero de 2007, aunque el sistema no tuvo un nombre público hasta que se liberó la primera versión del iPhone SDK. (Domínguez, 2014). iOS es un sistema operativo móvil de Apple desarrollado originalmente para el iPhone, siendo después usado en el iPod touch y en el iPad. En la actualidad este sistema operativo también se encuentra en Apple TV y en Smartwatch todos los productos originales de Apple.

iOS es un sistema operativo basado en Unix debido a que es una derivada del sistema operativo para computadores de Apple. A la vez MacOS está basado en el núcleo de Darwin BSD.

La arquitectura de iOS cuenta de las siguientes capas:

- **Core OS:** Esta capa brinda servicios de bajo nivel, es la única capa que controla el hardware del dispositivo, además de dar servicio de redes, estos servicios son basados en las instalaciones de la capa de kernel y controladores de cada dispositivo. (Apple Inc., 2017).
- **Core Services:** Core services es la capa de servicio centrales ya que bríndalos servicios necesarios para la ejecución de las aplicaciones, dejando de lado la interfaz de usuario. (Apple Inc., 2017). Esta capa está construida principalmente sobre los frameworks Core Foundation⁶, webKit⁷ y QuickLook⁸.
- **Media:** Brinda gráficos y multimedia de alta calidad, aprovechando las tecnologías de la capa media se puede incorporar gráficos en 2D y 3D, animaciones, audio y video. (Apple Inc., 2017)
- **Cocoa Touch:** Cocoa Touch está basado en el framework cocoa que permite desarrollar aplicaciones nativas para el sistema operativo ya que este framework incluye reconocimiento de gestos, animaciones y librerías para la interfaz. (Domínguez, 2014)

⁶ **Core Foundation:** Framework desarrollado por Appel Inc. que proporciona servicios de software fundamentales útiles para los servicios de aplicaciones.

⁷ **webKit:** motor de navegación web rápida.

⁸ **QuickLook:** Motor de gráficos 3D de Apple Inc.

1.2.3. Windows Phone

Este sistema móvil está desapareciendo ya que su creador ha dejado a un lado el proyecto, ya que no ha tenido tanta acogida como la competencia, por tal motivo no se toma en cuenta para el desarrollo de la aplicación y se dará solo una breve introducción al sistema. (Domínguez, 2014). Windows Phone es un sistema operativo móvil desarrollado por la empresa Microsoft, siendo así un sucesor del poco conocido Windows Mobile. Fue lanzado el 15 de febrero del 2010, con el fin de representar una alternativa a los sistemas operativos de la época como iOS, Android y Blackberry OS.

Los principales smartphones que utilizaban este sistema operativo eran los teléfonos de la marca Nokia, pero por la razón antes mencionada Nokia ha decidido lanzar sus nuevos productos con el sistema operativo Android.

1.2.4. Blackberry OS

Es un sistema operativo de código cerrado desarrollado por la empresa BlackBerry, lanzado por la misma en el año 1999. El soporte brindado por la compañía fue hasta el año 2013. (Domínguez, 2014). El proyecto fue abandonado por la falta de actualizaciones en sus productos y no soporto la competencia que ejercían los sistemas operativos de la época como iOS y Android.

1.3. Aplicaciones Móviles

Las aplicaciones móviles son pequeños programas informativos que se ejecutan sobre plataformas específicas como iOS y Android, que realizan operaciones simples. (Esteban Vázquez-Cano, 2015). Para ejecutar dichos programas no es necesaria una conexión a internet dependiendo de la aplicación.

La facilidad en su manejo, la rapidez de la respuesta y su instalación en un terminal que habitualmente llevamos siempre con nosotros (teléfono móvil) ha conseguido el gran auge de estas aplicaciones (Gordon, Rezzadeh, & Li, 2015). Dando como resultado una gran demanda de aplicaciones móviles para cada actividad que realizamos a lo largo del día.

Estés tipo de programas son realizado sobre diferentes lenguajes de programación y entornos de desarrollo diferentes, utilizando las herramientas propias dependiendo de cada sistema operativo en el que se desarrolla.

1.3.1. Tipos De Aplicaciones

Para la creación de una aplicación móvil existen varias formas para desarrollarlas. Cada tipo de aplicación tiene diferentes características que se deben tomar en cuenta antes de seguir una línea de desarrollo. (Montiel, 2017). Aunque el usuario final no note la diferencia, desde el punto de vista técnico se tiene limitaciones al momento de acceder al hardware dependiendo del tipo de aplicación y su línea de desarrollo.

- **Aplicaciones Nativas:** Las aplicaciones nativas son aquellas que se desarrollan con el SDK⁹ que proporciona cada sistema operativo, el mismo que tiene su propio lenguaje de programación en el caso de iOS que tiene Objective-C y para Android es Java. (Montiel, 2017). Para cada sistema operativo el SDK es diferente, por este motivo el desarrollo de estas aplicaciones es más costoso ya que se necesita tener desarrolladores experimentados para cada plataforma, pero se obtiene un rendimiento óptimo ya que se ejecutan en su propio entorno.

Además, para ser ejecutadas no necesitan de ningún complemento, dando así una mejor fluidez para el usuario, por otra parte, ya que están verdaderamente instaladas o integradas al dispositivo estas pueden hacer uso de todos sus componentes como cámaras, GPS¹⁰, acelerómetro entre otros. (Domínguez, 2014).

También ofrecen un diseño acorde al sistema operativo en el que se ejecuta, tomando la librería de gráficos propias de cada plataforma. Ofreciendo una mejor experiencia de usuario ya que estos están acostumbrados a la interfaz de su sistema operativo.

- **Aplicaciones Web:** “Se basa en contenido web, se ejecutan en un navegador y no hacen uso de las funcionalidades del dispositivo” (Domínguez, 2014). Este tipo de aplicaciones también llamadas WebApps, están escritas en HTML¹¹, JavaScript y CSS¹², como se sabe estas herramientas son utilizadas para desarrollo web. Al usar herramientas de desarrollo web no utilizan un SDK, permitiendo que los programadores poco experimentados puedan desarrollar aplicaciones para esta clase de dispositivos.

Las aplicaciones web necesitan internet ya que no están instaladas en el dispositivo, al contrario, usan navegadores para poder ser visualizadas y es necesario contar con un URL¹³ para acceder a ellas (Montiel, 2017). Al necesitar de un navegador ya instalado no

⁹ **SDK:** Software Development Kit

¹⁰ **GPS:** Global Positioning System

¹¹ **HTML:** Hypertext Transfer Protocol

¹² **CSS:** Cascading Style Sheets

¹³ **URL:** Localizador de recursos uniforme

permite que la aplicación aproveche el potencial del dispositivo, así como no poder utilizar al máximo todos los componentes del mismo.

Por ser desarrolladas con herramientas web su interfaz es independiente de la apariencia del sistema operativo, por este motivo las WebApps ofrecen una interacción reducida a comparación que con las aplicaciones nativas.

- **Aplicaciones Híbridas:** Las aplicaciones híbridas mezclan los dos tipos de aplicaciones anteriores, su desarrollo es muy parecido a las WebApps con la diferencia que al final se compila y se empaqueta como una aplicación nativa (Montiel, 2017).

Esto ayuda a reutilización del código ya que se puede usar el mismo código para aplicaciones para plataformas iOS como Android.

A diferencia de las aplicaciones Web, estas permiten al usuario utilizar los recursos del teléfono usando librerías, tal como lo haría una aplicación nativa.

Así como se combinan los métodos de desarrollo también se combina la interfaz de usuario dando así una mezcla de elementos en HTML, CSS y elementos propios de cada plataforma.

Con los tipos de aplicaciones antes mencionados ahora se puede analizar el tipo de aplicación es mejor para el proyecto que se necesite realizar.

1.4. Framework

Un framework está definido como un marco de trabajo que agrupa conceptos, prácticas y criterios de programación que serán utilizados en el desarrollo de una aplicación. Estos se basan en patrones de diseño generando la estructura de la aplicación. “Existen diferentes Framework que intentan unificar la mayoría de las aplicaciones partiendo de un lenguaje común y la tendencia debido a la tendencia de unificar el programador web con el programador de aplicaciones” (Contreras, 2016). El patrón de diseño MVC es uno de los más usados en la actualidad, siendo implementado por frameworks conocidos como: Ruby on Rails, Spring, Angular, Ionic, Symfony, etc.

Esto permite que los programadores se enfoquen en escribir código limpio, dejando de lado la estructura de la aplicación.

1.4.1. Ventajas De Utilizar Un Framework

- El programador no necesita plantear una estructura de la aplicación, ya que el framework proporciona un esqueleto que debe escribir el código.

- Existen varios foros y documentación para que el programador se puede guiar para resolver problemas al momento de realizar una aplicación.
- Es más fácil encontrar herramientas como librerías que han sido adaptadas a un framework en concreto para facilitar el desarrollo.

1.5. Framework React Native

React Native es un framework basado en JavaScript que sirve para el desarrollo de aplicaciones móviles de presentación nativa para iOS y Android. (React Native, 2017) Su núcleo se basa en React¹⁴, pero usa módulos nativos en lugar de módulos web como bloques de construcción.

Este framework usa una técnica que realiza llamadas asíncronas al sistema operativo donde se ejecuta la aplicación, que llama a las API¹⁵ de widget nativas. “Hay un motor de JavaScript, y la API de React es parecida a React para diseño de aplicaciones web. La diferencia es principalmente con el objetivo; en lugar de un DOM¹⁶, hay llamadas API asíncronas.” (Boduch, 2017, pág. 287).

Ahora los desarrolladores web pueden construir aplicaciones móviles que se ven como aplicaciones verdaderamente nativas usando una biblioteca de JavaScript. Además, debido a que el código en su gran mayoría sirve para otras plataformas dando como resultado un desarrollo simultáneo tanto para iOS como para Android.

Las aplicaciones son escritas en un formato híbrido entre JavaScript y XML¹⁷ conocido como JSX, este se comunica con las API's nativas de representación como Objective-C para iOS o Java para Android, por lo tanto, las aplicaciones procesarán componentes de UI móviles reales, también se comunican con los componentes del sistema dando así acceso a las características de la plataforma tales como la cámara, ubicación del GPS, entre otros.

Las ventajas de usar React Native se ven reflejadas en el rendimiento de la aplicación ya que usa las API's de renderizado estándar de la plataforma, funcionando por separado del subproceso de la interfaz de usuario, por lo que la aplicación obtiene un gran rendimiento a comparación de aplicaciones basadas en HTML y CSS. (React Native, 2017). Otra ventaja es que permite usar código nativo cuando se lo necesite ya que puede combinar los

¹⁴ **React:** Es una librería JavaScript focalizada en el desarrollo de interfaces de usuario. Esa es su principal área de trabajo, pero lo cierto es que con todo el ecosistema de aplicaciones y herramientas y componentes

¹⁵ **API:** interfaz de programación de aplicaciones

¹⁶ **DOM:** Modelo de Objetos del Documento

¹⁷ **XML:** Lenguaje de marcado

componentes escritos en Objective-C y Java, esta opción se utiliza cuando se necesita optimar una función especificando la plataforma para la que se está trabajando.

1.5.1. Componentes

“Estos componentes se crean con varias propiedades que son inmutables, una vez creado el componente no se pueden modificar, pero se puede modificar su estado” (React Native, 2017). Los estados de los componentes pueden ser modificados una vez se hayan creados los componentes, es recomendable que los componentes no tengan estado, ya que lo ideal es que no tengan estado y solo cuenten con propiedades.

1.5.2. JSX

Es un pseudolenguaje¹⁸ que facilita desarrollar tanto aplicaciones web como móviles, que permite escribir un lenguaje de marcado dentro del código. Se ve como HTML en la web, excepto que en lugar de elementos o etiquetas web como, <div> o , usa componentes React. (React Facebook, 2018). Como por ejemplo la etiqueta <Text> que es un componente incorporado que solo muestra texto.

También es utilizado el estándar conocido como ES6 o ES2015, es un conjunto de mejoras en JavaScript parte del estándar oficial, pero que aún no es compatible con todos los navegadores, por lo que a menudo no se usa aún en el desarrollo web. React Native soporta ES6, para que se pueda usar este estándar sin preocuparse por la compatibilidad en un futuro.

1.5.3. Data Flow de React Native

“El flujo en React Native es unidireccional. Manteniendo una jerarquía de componentes, en la que cada componente depende solo de su padre y su propio estado interno” (React Native, 2018). Cuando el estado de un componente cambia, el mismo y todos sus hijos son destruidos para dar paso a un nuevo componente que se crea con el método render. Este cambio fluye de arriba hacia abajo. Esto es implementado con el fin de mejorar el rendimiento.

Al momento de crear un nuevo componente se lo hace ejecutando código JavaScript, sin acceder al DOM. Una vez que se ejecuta la función render, React crea un nuevo DOM virtual con el contenido de los componentes nuevos creado bajo código JavaScript puro. Luego se compara el DOM virtual con el DOM real del navegador y se crea una lista con los cambios que se deben realizar para que el DOM real sea igual al DOM virtual.

¹⁸ **Pseudolenguaje:** Lenguaje de especificación de algoritmos

Finalmente se ejecutan los cambios de la lista, actualizando así el DOM real aplicando los mínimos cambios posibles.

- **DOM:** Es un estándar de W3C (World Wide Web Consortium). Es un estándar para acceder a los documentos. El Modelo de Objetos de Documentos es una plataforma y una interfaz de lenguaje neutral que permite que los programas y scripts accedan y actualicen dinámicamente el contenido, la estructura y el estilo de un documento. (w3schools, 2018).

1.5.4. Arquitectura Flux en React Native

Debido a que React se ocupa de la vista en MVC, Flux es un patrón de programación que se encargará de la parte de modelo. “Flux es la arquitectura de la aplicación que usa Facebook para crear aplicaciones web del lado del cliente. Complementa los componentes de vista de React utilizando un flujo de datos unidireccional” (Flux Facebook, 2018). Esta es una arquitectura que está diseñada en base al concepto de data Flow que maneja React ya que en esta arquitectura los datos viajan siempre en una única dirección acoplándose así a las necesidades de React Native.

- **Store:** Store o tienda es la lógica de la aplicación se encuentran en las Stores, el rol que desempeña es similar al modelo en una arquitectura MVC tradicional. (Facebook, 2018). Un store puede verse como una colección de N modelos, al mismo tiempo puede tener una o varias tiendas, dependiendo de cómo se organiza el código.
- **View:** React proporciona el tipo de vistas libremente renovables que necesitamos para la capa de visualización. Cerca de la parte superior de la jerarquía de vistas anidadas, un tipo especial de vista escucha los eventos que transmiten las stores de las que depende. (Facebook, 2018). A eso se le llama una vista de controlador, ya que proporciona el código para obtener los datos de las stores.
- **Dispatcher:** El despachador es quien gestiona todo el flujo de datos en la aplicación, es un registro con las devoluciones de llamadas en las tiendas que tiene un mecanismo simple para distribuir las acciones en las tiendas. (Facebook, 2018)
- **Action:** Es el operador que expone un método para él envió de datos a las stores, para la creación de una acción se puede envolver en un método de ayuda semántica que envíela acción al despachador.

La figura 1, presenta la estructura que genera el framework para crear una aplicación móvil.

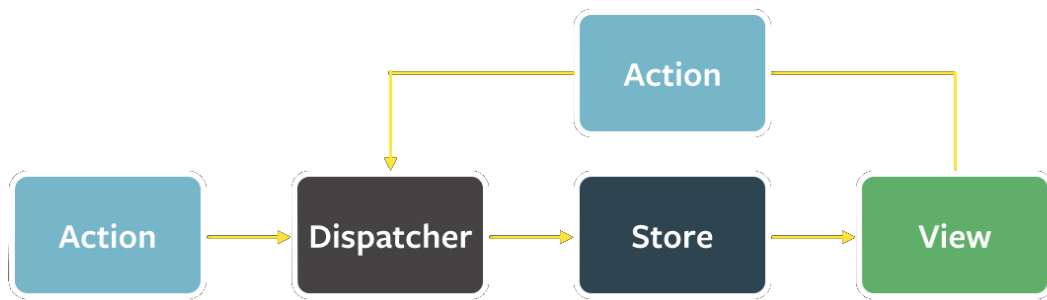


Fig.1 Arquitectura de aplicaciones generadas por el framework React Native
Fuente: Facebook GitHub

1.5.5. Instalación del Framework React Native

Para poder utilizar el framework se necesita primero una lista de requerimientos preinstalados para su perfecto funcionamiento tales como:

- NodeJs¹⁹
- Java
- Java SE Development kit
- Android Studio²⁰

Una vez instalado todos los requisitos se procede a la configuración del kit de desarrollo de software de Android o por sus siglas en inglés SDK.

Posterior a tener listo el entorno de trabajo se procede a instalar el framework React Native con la línea de comando “npm install -g react-native-cli” en el símbolo del sistema.

1.6. Framework Ionic

“El framework Ionic de código abierto cuenta con una rica biblioteca de componentes básicos y componentes de interfaz de usuario que facilitan el diseño de aplicaciones web y móviles de alto rendimiento utilizando tecnologías web como HTML, CSS y JavaScript” (Ionic, 2018). Ionic soporta HTML con una variedad de controles para la interfaz gráfica orientada a las aplicaciones móviles. Estos componentes que se adicionan están compuestos por HTML, CSS y JavaScript, con la particularidad de que se comportan de una forma similar a los controles nativos que se usan normalmente en las aplicaciones dependiendo de la plataforma, es decir simulan ser componentes nativos del sistema operativo.

¹⁹ **NodeJS:** es un entorno de ejecución para JavaScript

²⁰ **Android Studio:** ID de desarrollo de aplicaciones para Android

Ionic está basado en Apache Cordova y Angular, por lo que es necesario tener conocimientos básicos de estas herramientas para la creación de aplicaciones.

1.6.1. Apache Cordova

“Apache Cordova es un framework de desarrollo móvil de código abierto. Le permite utilizar tecnologías web estándar: HTML5, CSS3 y JavaScript para el desarrollo multiplataforma” (Apache Cordova, 2018) . Este framework transforma el lenguaje web en lenguaje nativo para dispositivos móviles con lo que se generan aplicaciones híbridas.

Cordova consta de tres componentes:

- **Código fuente:** Es un contenedor de aplicaciones nativas que tiene cada plataforma que se comunican con el hardware.
- **Conjunto de API's:** que proporcionan una aplicación web que se ejecuta dentro del acceso del contenedor a las capacidades del dispositivo nativo.
- **Conjunto de herramientas:** Son las herramientas utilizadas para administrar el proceso de creación de proyectos de aplicaciones. (Ravulavaru, 2015).

“La aplicación en sí misma se implementa como una página web, de manera predeterminada” (Apache Cordova, 2018). Cordova no solo junta la aplicación web con la aplicación nativa, sino también proporciona un conjunto de API escritas en JavaScript que interactúa con las características nativas del dispositivo como la cámara, sensores, bluetooth, etcétera.

Para crear una aplicación con Cordova primero se crea una aplicación web, después debe empaquetarse en un contenedor nativo, posteriormente se comienza en la fase de pruebas y depuración, por último, al ser una aplicación híbrida da la opción de distribuirla por una tienda de aplicaciones dependiendo de la plataforma.

Ya una vez creada la aplicación la interfaz de usuario consta de una sola pantalla que no contiene nada más que una vista web que ocupa el espacio disponible del dispositivo móvil. A medida que el usuario interactúa con la aplicación, los enlaces o códigos JavaScript se cargan desde los archivos empaquetados.

1.6.2. Angular 4

Es una plataforma que sirve para facilitar la creación del front-end para aplicaciones web, con este framework se busca librarse de incrustaciones de código back-end separando así la

funcionalidad de las otras capas. Integrando varias bibliotecas, algunas necesarias y otras opcionales. (Ingavélez, Hilera, & Timbi, 2016).

“Angular es un lenguaje estructurado para aplicaciones web dinámicas Javascript, orientado a operaciones CRUD (Create, Read, Update y Delete) mediante técnica AJAX.” (Ingavélez, Hilera, & Timbi, 2016). El enlace de datos de angular facilita introducir los valores de los datos en los controles HTML y convertir las respuestas de los usuarios en acciones y actualizaciones de los valores presentados.

1.6.3. Arquitectura de Aplicaciones Creadas con Ionic

“Ionic, al estar basado en Angular, utiliza el patrón conocido como Vista-Controlador que fue popularizado por frameworks como Cocoa Touch” (Gallego, 2017).

En este tipo de patrón las diferentes vistas se pueden dividir en vistas hijas, incluso podrían contener a su vez otras vistas hijas. Los controladores están enlazados a estas vistas y se encargan de proporcionar los datos necesarios y la funcionalidad de los diferentes componentes.

En la siguiente figura se presenta la arquitectura que maneja el framework Ionic para crear aplicaciones.

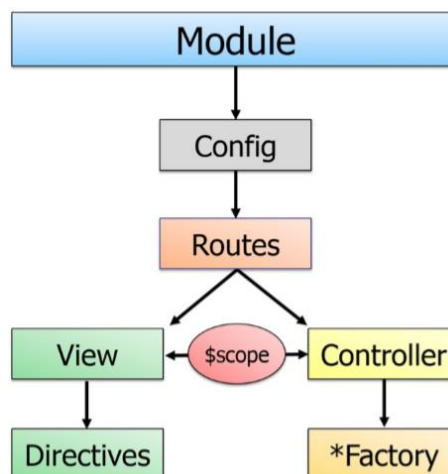


Fig 2. Arquitectura de aplicaciones generadas por el framework Ionic
Fuente: Manual de introducción a Ionic

1.6.4. Typescript

“TypeScript, es un lenguaje que se basa en el JavaScript moderno y agrega una comprobación de tipos estática” (Microsoft, 2018). Cuando escribe código TypeScript, puede usar una herramienta como el compilador para eliminar construcciones determinadas del tipo

y reescribir cualquier código. Por tanto, TypeScript amplía la sintaxis de JavaScript, pudiendo utilizar código de JavaScript y TypeScript conjuntamente sin problemas.

1.6.5. Instalación del Framework Ionic

Para poder utilizar el framework se necesita primero una lista de requerimientos preinstalados para su perfecto funcionamiento tales como:

- Node.js
- Java
- Java SE Development kit
- Android Studio
- Gradle

Una vez instalado todos los requisitos se procede a la configuración del kit de desarrollo de software de Android o por sus siglas en inglés SDK.

Posterior a tener listo el entorno de trabajo se procede a instalar el framework Ionic con la línea de comando “npm install -g ionic” en el símbolo del sistema.

1.7. Metodología XP

La metodología Programación Extrema o XP, es una metodología ágil para proyectos pequeños y de medio tamaño, basada en valores tales como: simplicidad, comunicación, retroalimentación, coraje y respeto.

“En la metodología extrema, todos los requisitos se expresan como escenarios (llamados historias de usuario), los cuales se implementan directamente como una serie de tareas” (Gómez, 2016). Se debe desarrollar cada tarea en pareja y se realizan pruebas para constatar su funcionamiento correcto, de acuerdo con los requerimientos del cliente.

1.7.1. Valores

- **Simplicidad:** La simplicidad es el fundamento principal de la programación extrema, en esta metodología se simplifica el diseño del software con el fin de agilizar el desarrollo y simplificar la documentación. “Un diseño simple siempre lleva menos tiempo para terminar que uno complejo. Así que siempre se hace lo más simple que podría funcionar” (Extreme Programming, 2018). Se intenta simplificar las funciones de la aplicación sin dejar de lado los requerimientos del cliente, lo que se simplifica es el método o la forma en que se codifica el algoritmo, mas no su funcionalidad.

Para simplificar la documentación se debe comentar el código, explicar para que sirve la variable o método en su nombre, generando así una auto documentación ayudando al siguiente programador a entender el código sin necesidad de acudir a la documentación.

- **Comunicación:** “La comunicación entre el cliente y el equipo permite que todos los detalles del proyecto sean tratados con la atención y agilidad que se merecen” (Laínez, 2014, pág. 118). La comunicación para los programadores no solo se da dialogando, sino también con un código simple y autodocumentado asignando nombres largos a variables y funciones que describan su funcionamiento, de esta forma se asegura que otro programador pueda entender el código para poder modificarlo.

La comunicación con el cliente debe ser constante ya que el cliente es parte esencial del equipo de desarrollo, es el que decide las características que tienen prioridad y siempre debe estar disponible para solucionar dudas.

- **Retroalimentación:** Como producto de la comunicación con el cliente se obtiene la opinión del cliente en tiempo real a medida que avanza el proyecto.

Al obtener resultado en ciclos cortos de tiempo, se minimiza el tener que rehacer funciones que no cumplen con las expectativas del cliente, para realizar modificaciones cuando sea necesario y no basarse en especulaciones que se pueden generar en el equipo. (Laínez, 2014)

- **Coraje:** El coraje consiste en tener fuerza de voluntad para programar para hoy y no dejar las cosas para el siguiente día. Se requiere valentía para desechar un código que ha pasado a ser obsoleto, sin importar el esfuerzo que se le ha invertido. (Extreme Programming, 2018).
- **Respeto:** El respeto es fundamental en cualquier grupo de trabajo, este consiste en respetar la codificación del compañero o hacer cambios que retrasen el avance del proyecto. Los miembros del equipo respetan el trabajo del resto no haciendo menos a otros, una mejor autoestima en el equipo eleva su ritmo de producción.

1.7.2. Características Fundamentales

Las características fundamentales del método son:

- **Desarrollo iterativo e incremental:** Ciclos cortos que generan mejoras constantemente.
- **Pruebas unitarias continuas:** Se realizan pruebas frecuentemente para comprobar la funcionalidad del código.

- **Programación en parejas:** Es recomendable que el desarrollo se lo haga en parejas, así el código es revisado y discutido mientras se lo escribe.
- **Frecuente integración:** El equipo de programación debe integrarse con el cliente o usuario. Se recomienda que un representante del cliente trabaje en conjunto con el equipo de desarrollo.
- **Corrección de errores:** Todos los errores deben ser corregidos antes de añadir nueva funcionalidad. Hacer entregas frecuentes.
- **Refactorización del código:** Se refactoriza el código reescribiendo ciertas partes para aumentar su legibilidad sin afectar su funcionalidad, para esto se usan las pruebas unitarias, confirmando así su correcto funcionamiento.
- **Propiedad del código compartida:** se promueve que todo el equipo de trabajo pueda corregir y extender cualquier parte del proyecto. (Laínez, 2014)

1.7.3. Roles

- **Programador:** Es la esencia del equipo, es quien escribe las pruebas unitarias y produce el código del sistema.
- **Cliente:** Es aquel que define las historias de usuario y las pruebas funcionales para validar su implementación. Asigna la prioridad a cada historia de usuario y decide cuáles se implementan en cada iteración.
- **Tester:** Ayuda al cliente a escribir las pruebas funcionales.
- **Tracker:** Es el que se encarga del seguimiento del proyecto y verifican el tiempo estimado con el tiempo realmente invertido.
- **Entrenador:** Es la persona responsable del proceso total. Guía a los miembros del equipo para seguir el proceso correctamente.
- **Consultor:** Es una persona externa al proyecto con un conocimiento específico en alguna área necesaria para el proyecto. Ayudando a resolver problemas que pueda suscitarse.
- **Gestor:** Es el dueño de la empresa y el lazo entre clientes y programadores. Su labor esencial es la coordinación. (Extreme Programming, 2018)

1.8. Norma ISO 9126

La ISO 9126 es una norma creada en 1992 por la Organización Internacional de la Normalización, es utilizada para evaluar la calidad de un software.

“Esta normativa se define por medio de seis principios fundamentales: funcionalidad, mantenibilidad, eficiencia, confiabilidad, usabilidad, portabilidad y una séptima que no es principal pero sí valorable, que es la calidad de uso” (Durán, 2015, pág. 28).

A continuación, se presentan los principios de la norma ISO 9126.



Fig 3. Características de la ISO 9126

Fuente: ISO 9126

1.8.1. Portabilidad

Es capacidad de un software para ser trasladado y adaptado desde una plataforma a otra.

TABLA 1.1 Sub-características de portabilidad

Adaptabilidad	La adaptabilidad de la define como la capacidad del software para adaptarse a diversos sistemas operáticos, así mismo como a diferentes dispositivos.
Capacidad de instalación	Esta característica se basa en que tan fácil es la instalación para el usuario final.
Capacidad de reemplazamiento	Es la característica del software para ser usado en lugar de otro software especificado en el ambiente de dicho software especificado.
Conformidad	Es la característica que reúne todas las características anteriores.

Nota. Adaptado de “Despliegue y puesta en funcionamiento de componentes software: UF1291”, Granado, R.,2014, IC Editorial, pág. 90.

1.8.2. Funcionalidad

Es un conjunto de atributos que posee el software para satisfacer las necesidades del cliente que adquiere el software.

TABLA 1.2 Sub-características de funcionalidad

Adecuación	Es la capacidad de cumplir tareas específicas. Atributos del software que cubren las necesidades del cliente para desempeñarse en el área que se lo necesita.
Precisión	Atributos del software que indican que su funcionamiento es correcto y realiza las funciones para las que fue diseñado.
Interoperabilidad	Atributos que debe poseer el software para interactuar con sistemas externos.
Seguridad	Atributos del software relacionados con su habilidad para prevenir acceso no autorizado ya sea accidental o deliberado, a programas y datos.

Nota. Adaptado de “Despliegue y puesta en funcionamiento de componentes software: UF1291”, Granado, R.,2014, IC Editorial, pág. 89.

1.8.3. Fiabilidad

Un conjunto de atributos relacionados con la capacidad del software de mantener su nivel de prestación bajo condiciones establecidas durante un período establecido.

TABLA 1.3 Sub-características de fiabilidad

Madurez	Es el conjunto de atributos del sistema que se han ido aumentando a lo largo del tiempo para evitar los fallos que pueden generarse por el mal uso del usuario o por problemas de hardware.
Recuperabilidad	Es la capacidad del sistema para recuperar información luego de un fallo.
Tolerancia a fallos	Propiedades que posee el software para seguir funcionando a pesar de sufrir fallos

Nota. Adaptado de “Despliegue y puesta en funcionamiento de componentes software: UF1291”, Granado, R.,2014, IC Editorial, pág. 89.

1.8.4. Usabilidad

Se denomina a la capacidad del producto para ser comprendido y utilizado de forma sencilla.

TABLA 1.4 Sub-características de usabilidad

Aprendizaje	Capacidad del software para enseñar al usuario la forma en que se debe utilizar cada función.
Claridad	“Es el conjunto de características que posee el sistema para ser entendido por el usuario. Para ello, no solo debe tener una buena interfaz, sino también disponer de todos los archivos y documentación necesaria para saber cómo utilizarlo correctamente” (Durán, 2015, pág. 32).
Operatividad	Conjunto de características del software que ayudan al usuario a utilizar el sistema.
Atracción	Se lo podría clasificar como el elemento estético que mejora la presentación del sistema sin dejar de lado su funcionalidad.

Nota. Adaptado de “Despliegue y puesta en funcionamiento de componentes software: UF1291”, Granado, R.,2014, IC Editorial, pág. 89.

1.8.5. Eficiencia

Se define como eficiencia del producto la forma de utilizarlo de manera correcta, de acuerdo con las especificaciones concretas para el propósito que fue diseñado.

TABLA 1.5 Sub-características de eficiencia

Comportamiento en el tiempo	Atributos del software que se relacionan con los tiempos de respuesta y procesamiento y en las tasas de rendimientos en desempeñar su función.
Comportamiento de recursos	Usar las cantidades y tipos de recursos adecuados cuando el software lleva a cabo su función bajo condiciones determinadas.

Nota. Adaptado de “Despliegue y puesta en funcionamiento de componentes software: UF1291”, Granado, R.,2014, IC Editorial, pág. 89.

1.8.6. Mantenibilidad

Es la capacidad del software para ser actualizado o editado para ser mejorado, alargando así su vida útil.

TABLA 1.6 Sub-características de mantenibilidad

Estabilidad	Es como el software reacciona frente a nuevas actualizaciones.
Facilidad de análisis	Es la forma en que el código está estructurado, en esa característica se evalúa, la facilidad con la que se puede encontrar fallas o detectar partes que han sido modificadas en versiones anteriores
Facilidad de cambio	Es la capacidad del software para ser modificado en base a la documentación y al diseño del sistema.
Facilidad de pruebas	Atributos del software afines con el esfuerzo necesario para aprobar el software modificado.

Nota. Adaptado de “Despliegue y puesta en funcionamiento de componentes software: UF1291”, Granado, R.,2014, IC Editorial, pág. 89.

CAPÍTULO II

Estudio Comparativo

Ionic y React Native son frameworks con poco tiempo en el mercado, sin embargo, los dos prestan características muy completas para el desarrollo de aplicaciones móviles ofreciendo así productos de calidad.

Los dos frameworks ayudan a desarrollar productos de similar calidad, pero se diferencian en varios aspectos como: el lenguaje de programación, el entorno de desarrollo, las plataformas que soportan, la documentación entre otros aspectos.

Por esta razón se ve necesario realizar el estudio comparativo con el fin de seleccionar la mejor herramienta que será utilizada en el desarrollo de la aplicación móvil. Con el fin de facilitar el proceso de creación, buscando así disminuir costos y tiempo en el desarrollo de ésta.

Se elaborará un módulo de la aplicación utilizando los frameworks Ionic y React Native; para determinar mediante la experiencia, la calificación de los parámetros de evaluación ofrecidos por la norma ISO 9126 para seleccionar la herramienta adecuada para el desarrollo de la aplicación.

2.1. Escala de Valoración

Una escala de valoración consiste en evaluar varias cualidades o características y exponer un resultado, indicando el grado de cumplimiento de cada una.

Esta escala de valoración se realizará de manera cualitativa determinando aspectos básicos de cada herramienta de desarrollo móvil, se utilizará la escala de Likert por ser una de las más usadas en los estudios investigativos.

La escala de Likert consiste en un conjunto de ítems presentados en forma de afirmaciones o juicios, para pedir la reacción de los participantes. Es decir, se presenta cada afirmación y se solicita al sujeto que externé su reacción eligiendo uno de los cinco puntos o categorías de la escala. (Hernández, Fernández, & Baptista, 2014)

Esta escala se forma a base de un eje de atributos declarados en forma afirmativa o negativa y un eje de valoración numérica, por lo general dicha escala cuenta con 5 niveles de valoración que va desde 1 que significa que no cumple con el parámetro de medición y 5 que significa que cumple totalmente con el parámetro.

TABLA 2.7 Ejemplo de la escala de Likert

Valor	Significado
1	Nunca
2	La mayoría de las veces no
3	Algunas veces sí, algunas veces no
4	La mayoría de las veces si
5	Siempre

Fuente: Metodología de la investigación

2.2. Parámetros de Evaluación

El estudio comparativo de los frameworks Ionic y React Native se elaborará en base a las métricas de la norma ISO 9126, con la finalidad de determinar cuál de los dos frameworks produce el mejor aplicativo móvil.

Se evaluará cada característica por separado usando la siguiente tabla con las métricas propuestas aplicando la escala de Likert que se encuentra en la Tabla 2.7.

TABLA 2.8 Ejemplo de tabla de evaluación

Funcionalidad	React Native	Ionic
Adecuación	¿Puede el software desempeñar las tareas requeridas?	
Precisión	¿El resultado es el esperado?	
Interoperabilidad	¿El sistema puede interactuar con otro?	
Seguridad	¿El sistema impide el acceso no autorizado?	
TOTAL		
PROMEDIO		

Fuente: Propia

De esta forma se obtendrá los valores que determinaran cuál de los frameworks es el indicado para desarrollar el aplicativo que solicita la empresa. Debido a que estos parámetros están basados en la norma ISO 9126 se busca que el producto final tenga altos niveles de calidad.

2.3. Análisis Comparativo

2.3.1. Portabilidad

TABLA 2.9 Análisis de Portabilidad

Portabilidad		React Native	Ionic
Adaptabilidad	¿El software se puede trasladar a otros ambientes?	5	5
Capacidad de instalación	¿El software se puede instalar fácilmente?	4	4
Capacidad para reemplazar	¿El software puede reemplazar fácilmente otro software?	2	4
Conformidad	¿El software cumple con los estándares de Transportabilidad?	4	5
TOTAL		15	18
PROMEDIO		3,75	4,5

Fuente: Propia

Análisis:

- a) **Adaptabilidad:** Tanto Ionic como React Native se pueden trasladar a diferentes ambientes, siempre y cuando se cumplan con los requisitos que se piden en las instalaciones de cada framework, esto es posible para las tres grandes plataformas como Windows, Mac OS y Linux.
- b) **Capacidad de instalación:** Los dos frameworks para su instalación y perfecto funcionamiento dependen de varios programas y frameworks como Node.js, Android Studio o Xcode entre otros. Por este motivo la instalación de ambos frameworks tiene un pequeño grado de dificultad.
- c) **Capacidad para reemplazar:** React Native no se puede reemplazar fácilmente por otro software ya que para esto se necesitaría volver a escribir código, debido a que utiliza un lenguaje de programación propio. En cambio, Ionic tiene una pequeña compatibilidad

con Phonegap que permite usar pequeñas partes de Ionic para implementar en una aplicación realizada con Phonegap, además de estar basado en Angular que admite usar su código y ser implementado en una página web.

- d) **Conformidad:** En este caso Ionic tiene un óptimo funcionamiento en los tres sistemas operativos móviles más usados por los desarrolladores, en cambio React Native esta optimizado para funcionar en Mac Os dejando de lado varias funciones para las demás plataformas.

2.3.2. Funcionalidad

TABLA 2.10 Análisis de Funcionalidad

Funcionalidad		React Native	Ionic
Conveniencia	¿Puede el software desempeñar las áreas requeridas?	5	5
Precisión	¿El resultado es el esperado?	3	5
Interoperabilidad	¿El sistema puede interactuar con otro?	5	5
Seguridad	¿El sistema impide el acceso no autorizado?	2	2
TOTAL		15	17
PROMEDIO		3,75	4,25

Fuente: Propia

Análisis:

- a) **Adecuación:** Ambos frameworks cumplen con su cometido son opciones para desarrollar aplicaciones móviles híbridas con las mismas características que proporcionan las aplicaciones nativas.
- b) **Precisión:** Los dos frameworks dan como resultado una aplicación híbrida que puede ser publicada en las tiendas oficiales de las plataformas, con el caso que React Native no da soporte para crear aplicaciones para Windows Phone, Firefox OS, Caso contrario el framework Ionic si brinda soporte a las plataformas mencionadas anteriormente.
- c) **Interoperabilidad:** Tanto React Native como Ionic son capaces de interactuar con otros sistemas al igual que implementar varias plataformas a sus aplicaciones, tales como:

realizar pagos por PayPal, visualización de videos de Youtube, además de conectarse a varias bases de datos por medio de servicios Rest

- d) **Seguridad:** Ninguno de los frameworks ofrece seguridad para proteger los proyectos, es decir que cualquier persona que tenga acceso al computador puede visualizar, copiar, editar y eliminar los proyectos creados con estos dos frameworks. Después de empaquetar las aplicaciones y generar el archivo para ser publicado se aplican niveles de seguridad bajos para que la aplicación no pueda revelar su código fuente.

2.3.3. Fiabilidad

TABLA 2.11 Análisis de Fiabilidad

Fiabilidad		React Native	Ionic
Madurez	¿Muchas de las fallas han sido eliminadas durante el tiempo?	5	5
Tolerancia a las fallas	¿El software es capaz de manejar errores?	5	5
TOTAL		10	10
PROMEDIO		5	5

Fuente: Propia

Análisis:

- a) **Madurez:** Ionic después de cinco años de su lanzamiento ha mejorado en cada aspecto que lo conforman, desde su primera versión en el 2015 ha decidido implementar Angular 1 con el paso del tiempo y debido a requerimientos de los usuarios se han visto obligados a mejorar este aspecto, así que implementaron Angular 2 a partir de ese momento se han expandido los horizontes llegando así a soportar más plataformas como iOS y Whindows Phone, al igual que han ido corrigiendo pequeños errores que se han ido presentando a lo largo de sus actualizaciones.

Por otro lado, React Native desde su lanzamiento en junio del 2015 hasta los inicios del 2018 ha tenido cincuenta y una actualizaciones corrigiendo errores y aumentando funcionalidades en cada nueva versión del framework.

- b) **Tolerancia a fallas:** Los dos Frameworks son conocidos por su detección de errores en caliente, mejorando así la experiencia de los usuarios al momento del desarrollo de aplicaciones, cada uno presenta los errores en una consola al momento de ejecutar la compilación de los proyectos.

2.3.4. Usabilidad

TABLA 2.12 Análisis de Usabilidad

Usabilidad		React Native	Ionic
Claridad	¿El usuario comprende fácilmente como usar el sistema?	5	5
Capacidad de aprendizaje	¿Puede el usuario aprender fácilmente a utilizar el sistema?	3	4
Operatividad atractiva	¿El usuario puede utilizar el sistema sin mucho esfuerzo?	4	4
	¿La interfaz se ve bien?	2	2
TOTAL		14	15
PROMEDIO		3,5	3,75

Fuente: Propia

Análisis:

a) Claridad: En los dos casos el usuario puede aprender a usar el framework usando la extensa documentación que posee al igual que puede usar las aplicaciones móviles que ayudan con el aprendizaje de las dos herramientas, también se puede usar las varias ayudas que ofrecen los mismos usuarios ya que al tener gran popularidad ha provocado que se generen tutoriales en diversas plataformas como Youtube y GitHub, sin contar la gran cantidad de foros donde se ayudan a resolver problemas que se dan a lo largo del desarrollo.

b) Capacidad de Aprendizaje: Ionic es un framework que a pesar de su documentación es una herramienta que usa una mezcla de varias herramientas que hacen que el aprendizaje no sea tan rápido a comparación de otras herramientas, por otro lado, también usa herramientas conocidas por la mayoría de los desarrolladores web como: Angular y TypeScript, CSS, HTML y JavaScript que facilita el aprendizaje del framework.

React Native sin embargo de usar HTML, CSS y JavaScript también usa React una librería de JavaScript creada por Facebook que el usuario debe conocer previamente para poder desarrollar aplicaciones, por este motivo el aprendizaje de esta librería dificulta el aprendizaje total del framework.

c) Operatividad Atractiva: Ambos frameworks no cuentan con ninguna interfaz gráfica ni con un ID propio del framework que ayude al desarrollo de aplicaciones por tal motivo el

usuario tiene que buscar un editor de texto para poder escribir el código, sin embargo, los dos frameworks cuentan con pantallas para poder ver la aplicación compilada en tiempo real permitiendo así ver el comportamiento y los cambios realizados a lo largo de la realización del proyecto.

2.3.5. Eficiencia

TABLA 2.13 Análisis de Eficiencia

Eficiencia		React Native	Ionic
Comportamiento del tiempo	¿Qué tan rápido responde el sistema?	4	5
TOTAL		4	5
PROMEDIO		4	5

Fuente: Propia

Análisis:

- a) **Comportamiento del tiempo:** En este aspecto Ionic reacciona mucho más rápido, al realizar las mismas tareas que realiza React Native tales como: crear proyectos, desplegar aplicaciones, y determinación de errores.

2.3.6. Mantenibilidad

Este parámetro no ha sido determinado, debido a que no se puede realizar modificaciones en el código fuente, y comprobar la calidad del software en este aspecto.

2.4. Determinación del Software Ganador

Terminado el proceso de análisis basado en la norma ISO 9126, se ha logrado determinar el software ganador, para ser usado en el desarrollo de la aplicación móvil. A partir del análisis se ha generado una tabla con los puntajes obtenidos en cada uno de los parámetros evaluados.

A continuación, se muestra la tabla con las calificaciones, demostrando así el software ganador con mejores atributos de calidad, con eso se busca dar una solución eficiente y eficaz del problema.

TABLA 2.14 Análisis del Software Ganador

Parámetros ISO 9126	React Native	Ionic
Portabilidad	3,75	4,5
Funcionalidad	3,75	4,25
Fiabilidad	5	5
Usabilidad	3,5	3,75
Eficiencia	4	5
TOTAL	4	4,7

Fuente: Propia

Como resultado del análisis tenemos la calificación para cada uno de los parámetros propuestos, determinando como software ganador al Framework Ionic, que obtuvo una calificación de 4,70 sobre 5, teniendo una diferencia a favor de 0,70 puntos del otro framework competidor React Native.

CAPÍTULO III

Desarrollo Del Sistema

En el proceso de desarrollo del trabajo de grado se definió la metodología XP como la metodología a usarse en la implementación de la aplicación, para poder así utilizar las buenas prácticas y técnicas que la metodología ofrece, Obteniendo así una comunicación entre el cliente y el desarrollador, cumpliendo igualmente los objetivos marcados a corto plazo.

3.1. Planificación del Proyecto.

3.1.1. Roles de Usuarios

Se definen los roles de los usuarios que ayudará a distinguir a las personas involucradas en el desarrollo del aplicativo.

TABLA 3.15 Descripción de roles de usuarios

Nombre	Descripción	Rol
Msc. Daysi Imbaquingo	Encargada de revisiones de los avances del sistema.	Consultor
José Orbe	Encargado de pruebas funcionales.	Cliente
Robinson Ruano	Encargado del desarrollo del sistema.	Programador

Fuente: Propia

3.1.2. Tipos de Usuarios

- **Usuario Administrador:** Permite controlar el módulo administrativo del sistema y visualizar pedidos.
- **Usuario Cliente:** En este módulo los usuarios registraran datos básicos y tendrá las opciones de realizar pedidos a domicilios.

3.1.3. Historias de Usuarios

A continuación, se presenta las historias de usuarios que se han generado en las distintas reuniones con el gerente de la empresa “La Lico”

TABLA 3.16 Historia de usuario 01

Historia de Usuario	
Numero: 01	Usuario: Administrador
Nombre Historia: Acceso al Sistema web	
Prioridad: Alta	Riesgo en Desarrollo: Medio
Programador: Sr. Robinson Ruano	Iteraciones: 1
Descripción: Los Administradores pueden ingresar al sistema desde cualquier navegador.	

Fuente: Propia

TABLA 3.17 Historia de usuario 02

Historia de Usuario	
Numero: 02	Usuario: Administrador
Nombre Historia: Gestión del Portal	
Prioridad: Alta	Riesgo en Desarrollo: Medio
Programador: Sr. Robinson Ruano	Iteraciones: 1
Descripción: El administrador podrá gestionar usuarios repartidores y clientes, agregar y eliminar productos y promociones que se mostraran en la aplicación.	

Fuente: Propia

TABLA 3.18 Historia de usuario 03

Historia de Usuario	
Numero: 03	Usuario: Administrador
Nombre Historia: Visualización de Pedidos	
Prioridad: Alta	Riesgo en Desarrollo: Bajo
Programador: Sr. Robinson Ruano	Iteraciones: 2
Descripción: EL administrador podrá visualizar todos los pedidos que han sido registrados, con toda la información necesaria para realizar la entrega.	

Fuente: Propia

TABLA 3.19 Historia de usuario 04

Historia de Usuario	
Numero: 04	Usuario: Cliente
Nombre Historia: Acceso a la Aplicación Cliente	
Prioridad: Alta	Riesgo en Desarrollo: Medio
Programador: Sr. Robinson Ruano	Iteraciones: 1
Descripción: Los clientes pueden ingresar a la aplicación desde cualquier dispositivo Android o iOS.	
Fuente: Propia	

TABLA 3.20 Historia de usuario 05

Historia de Usuario	
Numero: 05	Usuario: Cliente
Nombre Historia: Visualización de Menú	
Prioridad: Alta	Riesgo en Desarrollo: Medio
Programador: Sr. Robinson Ruano	Iteraciones: 1
Descripción: Los clientes que accedan a la aplicación podrán visualizar y escoger las opciones presentadas.	
Fuente: Propia	

TABLA 3.21 Historia de usuario 06

Historia de Usuario	
Numero: 06	Usuario: Cliente
Nombre Historia: Visualizar Productos y Promociones	
Prioridad: Alta	Riesgo en Desarrollo: Bajo
Programador: Sr. Robinson Ruano	Iteraciones: 2
Descripción: El cliente podrá visualizar todos productos y promociones.	
Fuente: Propia	

TABLA 3.22 Historia de usuario 07

Historia de Usuario	
Numero: 07	Usuario: Cliente
Nombre Historia: Realizar Pedidos	
Prioridad: Alta	Riesgo en Desarrollo: Medio
Programador: Sr. Robinson Ruano	Iteraciones: 2
Descripción: El cliente podrá seleccionar los productos y promociones que desea adquirir.	

Fuente: Propia

3.1.4. Plan de Entregas

En esta fase se prosigue a ordenar en orden cronológico las historias de usuarios, para poder realizar un plan de entregas que debe ser cumplido en las iteraciones propuestas.

Los tiempos de desarrollo se medirán de acuerdo con la siguiente tabla:

TABLA 3.23 Equivalencias de tiempo de desarrollo

Tiempo	Equivalencia
Un día	Ocho horas

Fuente: Propia

TABLA 3.24 Plan de entregas

Módulos	Nro.	Historias de Usuarios	Fechas Estimadas	Esfuerzo		Iteraciones		Entregas	
				Días	Horas	1	2	1	2
Módulo de administración	1	Acceso al Sistema web	19/06/2018 a 21/06/2018	3	24	X		X	
	2	Gestión del Portal	22/06/2018 a 27/06/2018	4	32	X		X	
	3	Visualización de Pedidos	28/06/2018 a 02/07/2018	3	24	X		X	
Módulo de Cliente	4	Acceso a la Aplicación Cliente	03/07/2018 a 05/07/2018	3	24		X		X
	5	Visualización de Menú	06/07/2018 a 10/07/2018	3	24		X		X
	6	Visualizar Productos y Promociones	11/07/2018 a 16/07/2018	4	32		X		X
	7	Realizar un pedido	17/07/2018 a 20/07/2018	4	32		X		X

Fuente: Propia

3.1.5. Módulos Del Sistema

- **Módulo de Administración:** El administrador podrá acceder al módulo desde cualquier lugar y dispositivo que cuente con acceso a internet. Para gestionar los productos que se muestran en la aplicación móvil.
- **Módulo de Cliente:** La aplicación estará disponible para descargar desde la pagina oficial de “La Lico”. además, el usuario podrá acceder a un menu donde tendrá la opción de ver los productos, agregarlos al carrito y realizar la compra.

3.1.6. Planificación: Iteración I

a) Cornograma

TABLA 3.25 Cronograma de Iteración I

Nro.	Historia de Usuario	Fecha Estimada	Duración	
			Días	Horas
1	Acceso al Sistema Web	19/06/2018 a 21/06/2018	3	24
2	Gestión del Portal	22/06/2018 a 27/06/2018	4	32
3	Visualización de Pedidos	28/06/2018 a 02/07/2018	3	24
TOTAL			10	80

Fuente: Propia

b) Tareas de Historia 1: Acceso al Sistema

TABLA 3.26 Tareas de historia 1

Nro.	Nombre	Tiempo Estimado
1	Configuración del entorno de programación	8 horas
2	Configuración del servidor	6 horas
3	Diseño de interfaz gráfica, aplicación de plantilla	5 horas
4	Implementación de módulo de login	5 horas
TOTAL		24 horas

Fuente: Propia

- **Configuración del entorno de Programación**

TABLA 3.27 Tareas de historia 1 Tarea 1 configuración del entorno de Programación

Tarea	
Numero Tarea: 1	Numero Historia: 1
Nombre Tarea: Configuración del entorno de Programación	
Tipo de Tarea: General	Tiempo Estimado: 8 horas
Fecha Inicio: 19/06/2018	Fecha Fin: 19/06/2018
Programador Responsable: Robinson Ruano	
Descripción: Se procede a la configuración del ambiente de trabajo para el desarrollo de la aplicación.	
Componentes Para Implementar:	
Gestor de Texto: Visual Code	
Gestor de Base de Datos: MySQL	
CSM: Joomla	

Fuente: Propia

- **Configuración del servidor**

TABLA 3.28 Tareas de historia 1 Tarea 2 configuración del servidor

Tarea	
Numero Tarea: 2	Numero Historia: 1
Nombre Tarea: Configuración del servidor	
Tipo de Tarea: General	Tiempo Estimado: 6 horas
Fecha Inicio: 20/06/2018	Fecha Fin: 20/06/2018
Programador Responsable: Robinson Ruano	
Descripción: Se procede a configurar el servidor de la aplicación web con PHP y apache.	

Fuente: Propia

- **Diseño de interfaz gráfica, aplicación de plantilla**

TABLA 3.29 Tareas de historia 1 Tarea 3 diseño de interfaz gráfica, aplicación de plantilla

Tarea	
Numero Tarea: 3	Numero Historia: 1
Nombre Tarea: Diseño de interfaz gráfica, aplicación de plantilla	
Tipo de Tarea: Diseño	Tiempo Estimado: 5 horas
Fecha Inicio: 20/06/2018	Fecha Fin: 21/06/2018
Programador Responsable: Robinson Ruano	
Descripción: Se crea una interfaz simple para la ventana de login del sistema web del administrador utilizando una plantilla de Joomla.	

Fuente: Propia

- **Implementación de módulo de login**

TABLA 3.30 Tareas de historia 1 Tarea 4 implementación del módulo de login

Tarea	
Numero Tarea: 4	Numero Historia: 1
Nombre Tarea: Implementación de módulo de login	
Tipo de Tarea: Programación	Tiempo Estimado: 5 horas
Fecha Inicio: 21/06/2018	Fecha Fin: 21/06/2018
Programador Responsable: Robinson Ruano	
Descripción: Se implementan las funciones del módulo de login para que el usuario pueda acceder al portal con su usuario y contraseña.	

Fuente: Propia

c) **Tareas de Historia 2: Gestion del portal**

TABLA 3.31 Tareas de historia 2

Nro.	Nombre	Tiempo Estimado
1	Configuración del entorno de programación	8 horas
2	Diseño de botones para cada opción	3 horas
3	Implementación de CRUD para productos, sucursales y ciudades	12 horas
4	Implementar página de estadísticas para el administrador	9 horas
TOTAL		32 horas

Fuente: Propia

- **Diseño de interfaz gráfica del portal**

TABLA 3.32 Tareas de historia 2 Tarea 1 diseño de interfaz gráfica del portal

Tarea	
Numero Tarea: 1	Numero Historia: 2
Nombre Tarea: Diseño de interfaz gráfica del portal	
Tipo de Tarea: Diseño	Tiempo Estimado: 8 horas
Fecha Inicio: 22/06/2018	Fecha Fin: 22/06/2018
Programador Responsable: Robinson Ruano	
Descripción: Se realizará un diseño que sea simple y atractivo para el administrador. Que cuente con todas las opciones que necesite el usuario, editando el CSS de la plantilla previamente instalada.	

Fuente: Propia

- **Diseño de botones para cada opción**

TABLA 3.33 Tareas de historia 2 Tarea 2 diseño de botones para cada opción

Tarea	
Numero Tarea: 2	Numero Historia: 2
Nombre Tarea: Diseño de botones para cada opción	
Tipo de Tarea: Diseño	Tiempo Estimado: 3 horas
Fecha Inicio: 25/06/2018	Fecha Fin: 25/06/2018
Programador Responsable: Robinson Ruano	
Descripción: Se diseñan botones para el menú de usuario que direccionen a las opciones que se implementaran futuramente.	

Fuente: Propia

- **Implementación de CRUD para productos, sucursales y ciudades**

TABLA 3.34 Tareas de historia 2 Tarea 3 Implementación de CRUD para productos, sucursales y ciudades

Tarea	
Numero Tarea: 3	Numero Historia: 2
Nombre Tarea: Implementación de CRUD para productos, sucursales y ciudades	
Tipo de Tarea: Programación	Tiempo Estimado: 12 horas
Fecha Inicio: 25/06/2018	Fecha Fin: 27/06/2018
Programador Responsable: Robinson Ruano	
Descripción: La tarea describe implementar las funciones de administrador como gestión de productos, sucursales y ciudades.	

Fuente: Propia

- **Implementar página de estadísticas para el administrador**

TABLA 3.35 Tareas de historia 2 Tarea 4 Implementar página de estadísticas para el administrador

Tarea	
Numero Tarea: 4	Numero Historia: 2
Nombre Tarea: Implementar página de estadísticas para el administrador	
Tipo de Tarea: Programación	Tiempo Estimado: 12 horas
Fecha Inicio: 28/06/2018	Fecha Fin: 02/07/2018
Programador Responsable: Robinson Ruano	
Descripción: La tarea describe implementar la página de estadísticas para visualización del administrador.	

Fuente: Propia

d) Tareas de Historia 3: Vizualización de pedidos

TABLA 3.36 Tareas de historia 3

Nro.	Nombre	Tiempo Estimado
1	Diseño del módulo de Visualización de Pedidos	8 horas
2	Diseño de botones para cada opción	8 horas
3	Implementación de funciones principales	8 horas
TOTAL		24 horas

Fuente: Propia

- **Diseño del módulo de Visualización de Pedidos**

TABLA 3.37 Tareas de historia 3 Tarea 1 Diseño del módulo de Visualización de Pedidos

Tarea	
Numero Tarea: 1	Numero Historia: 3
Nombre Tarea: Diseño del módulo de Visualización de Pedidos	
Tipo de Tarea: Diseño	Tiempo Estimado: 8 horas
Fecha Inicio: 03/07/2018	Fecha Fin: 03/07/2018
Programador Responsable: Robinson Ruano	
Descripción: Se realizará un diseño que sea simple y atractivo para el cliente. Que cuente con todas las opciones que necesite el usuario.	

Fuente: Propia

- **Diseño de botones para cada opción**

TABLA 3.38 Tareas de historia 3 Tarea 2 Diseño de botones para cada opción

Tarea	
Numero Tarea: 2	Numero Historia: 3
Nombre Tarea: Diseño de botones para cada opción	
Tipo de Tarea: Diseño	Tiempo Estimado: 8 horas
Fecha Inicio: 04/07/2018	Fecha Fin: 04/07/2018
Programador Responsable: Robinson Ruano	
Descripción: Se diseñan botones para el menú de usuario que direccionen a las opciones que se implementaran futuramente.	

Fuente: Propia

- **Implementación de funciones principales**

TABLA 3.39 Tareas de historia 3 Tarea 3 Implementación de funciones principales

Tarea	
Numero Tarea: 3	Numero Historia: 3
Nombre Tarea: Implementación de funciones principales	
Tipo de Tarea: Programación	Tiempo Estimado: 8 horas
Fecha Inicio: 05/07/2018	Fecha Fin: 05/07/2018
Programador Responsable: Robinson Ruano	
Descripción: La tarea describe implementar las funciones, para poder visualizar los pedidos entrantes y tener un historial de todos los pedidos realizados.	

Fuente: Propia

3.1.7. Planificación: Iteración II

a) Cornograma

TABLA 3.40 Cronograma iteración II

Nro.	Historia de Usuario	Fecha Estimada	Duración	
			Días	Horas
1	Acceso a la aplicación	03/07/2018 a 05/07/2018	3	24
2	Visualización del Menú	06/07/2018 a 10/07/2018	3	24
3	Visualizar Productos y Promociones	11/07/2018 a 16/07/2018	4	32
4	Realizar Pedidos	17/07/2018 a 20/07/2018	4	32
TOTAL			11	112

Fuente: Propia

b) **Tareas de Historia 4: Acceso a la aplicación**

TABLA 3.41 Tareas de historia 4

Nro.	Nombre	Tiempo Estimado
1	Configuración del entorno de programación	12 horas
2	Diseño de interfaz gráfica del logan	4 horas
3	Implementación del código de verificación de acceso	8 horas
TOTAL		24 horas

Fuente: Propia

- **Configuración del entorno de programación**

TABLA 3.42 Tareas de historia 4 Tarea 1 Configuración del entorno de programación

Tarea	
Numero Tarea: 1	Numero Historia: 4
Nombre Tarea: Configuración del entorno de programación	
Tipo de Tarea: General	Tiempo Estimado: 12 horas
Fecha Inicio: 03/07/2018	Fecha Fin: 04/07/2018
Programador Responsable: Robinson Ruano	
Descripción: Se produce a la configuración del ambiente de trabajo para el desarrollo de la aplicación.	
Componentes Por Implementar:	
Gestor de Texto: Visual Code	
ID: Android Studio	
Framework: Ionic	

Fuente: Propia

- **Diseño de interfaz gráfica del login**

TABLA 3.43 Tareas de historia 4 Tarea 2 Diseño de interfaz gráfica del login

Tarea	
Numero Tarea: 2	Numero Historia: 3
Nombre Tarea: Diseño de interfaz gráfica del login	
Tipo de Tarea: Diseño	Tiempo Estimado: 4 horas
Fecha Inicio: 04/07/2018	Fecha Fin: 04/07/2018
Programador Responsable: Robinson Ruano	
Descripción: Se crea una interfaz simple para la ventana de login de la aplicación del cliente.	

Fuente: Propia

- **Implementación del código de verificación de acceso**

TABLA 3.44 Tareas de historia 4 Tarea 3 Implementación del código de verificación de acceso

Tarea	
Numero Tarea: 3	Numero Historia: 3
Nombre Tarea: Implementación del código de verificación de acceso	
Tipo de Tarea: Programación	Tiempo Estimado: 8 horas
Fecha Inicio: 02/07/2018	Fecha Fin: 02/07/2018
Programador Responsable: Robinson Ruano	
Descripción: Se implementan las funciones para que el usuario pueda acceder al menú de la aplicación móvil.	

Fuente: Propia

c) **Tareas de Historia 5: Vizualización de Menú**

TABLA 3.45 Tareas de historia 5

Nro.	Nombre	Tiempo Estimado
1	Diseño de interfaz gráfica del menú	9 horas
2	Implementación de animación y transiciones	7 horas
3	Implementación de funcionalidad del menú	8 horas
TOTAL		24 horas

Fuente: Propia

- **Diseño de interfaz gráfica del menú**

TABLA 3.46 Tareas de historia 5 Tarea 1 Diseño de interfaz gráfica del menú

Tarea	
Numero Tarea: 1	Numero Historia: 5
Nombre Tarea: Diseño de interfaz gráfica del menú	
Tipo de Tarea: Diseño	Tiempo Estimado: 9 horas
Fecha Inicio: 06/07/2018	Fecha Fin: 09/07/2018
Programador Responsable: Robinson Ruano	
Descripción: Se diseña la interfaz gráfica del menú de manera minimalista y funcional, con todas las opciones principales que contara la aplicación	

Fuente: Propia

- **Implementación de animación y transiciones**

TABLA 3.47 Tareas de historia 5 Tarea 2 Implementación de animación y transiciones

Tarea	
Numero Tarea: 2	Numero Historia: 5
Nombre Tarea: Implementación de animación y transiciones	
Tipo de Tarea: Diseño	Tiempo Estimado: 7 horas
Fecha Inicio: 09/07/2018	Fecha Fin: 09/07/2018
Programador Responsable: Robinson Ruano	
Descripción: Modificar la presentación con animaciones para botones del menú, también se incluirán transiciones al cambiar de una pantalla a la otra.	

Fuente: Propia

- **Implementación de funcionalidad del menú**

TABLA 3.48 Tareas de historia 5 Tarea 3 Implementación de funcionalidad del menú

Tarea	
Numero Tarea: 3	Numero Historia: 5
Nombre Tarea: Implementación de funcionalidad del menú	
Tipo de Tarea: Programación	Tiempo Estimado: 8 horas
Fecha Inicio: 10/07/2018	Fecha Fin: 11/07/2018
Programador Responsable: Robinson Ruano	
Descripción: La tarea describe implementar las funciones del menú para direccionar a todas las vistas de la aplicación.	

Fuente: Propia

d) Tareas de Historia 6: Visualización de Productos y Promociones

TABLA 3.49 Tareas de historia 6

Nro.	Nombre	Tiempo Estimado
1	Diseño del módulo de Visualización de Productos	8 horas
2	Diseño de botones para cada opción	4 horas
3	Creación de Web Service Rest	8 horas
4	Implementación de funciones principales	12 horas
TOTAL		32 horas

Fuente: Propia

- **Diseño del módulo de Visualización de Productos**

TABLA 3.50 Tareas de historia 6 Tarea 1 Diseño del módulo de Visualización de Productos

Tarea	
Numero Tarea: 1	Numero Historia: 6
Nombre Tarea: Diseño del módulo de Visualización de Productos	
Tipo de Tarea: Diseño	Tiempo Estimado: 8 horas
Fecha Inicio: 11/07/2018	Fecha Fin: 11/07/2018
Programador Responsable: Robinson Ruano	
Descripción: Se realizará un diseño que sea simple y atractivo para el cliente. Que cuente con todas las opciones que necesite el usuario.	

Fuente: Propia

- **Diseño de botones para cada opción**

TABLA 3.51 Tareas de historia 6 Tarea 2 Diseño de botones para cada opción

Tarea	
Numero Tarea: 2	Numero Historia: 6
Nombre Tarea: Diseño de botones para cada opción	
Tipo de Tarea: Diseño	Tiempo Estimado: 4 horas
Fecha Inicio: 12/07/2018	Fecha Fin: 12/07/2018
Programador Responsable: Robinson Ruano	
Descripción: Se diseñan botones para el menú de usuario que direccionen a las opciones que se implementaran futuramente.	

Fuente: Propia

- **Creación de Web Service Rest**

TABLA 3.52 Tareas de historia 6 Tarea 3 Creación de Web Service Rest

Tarea	
Numero Tarea: 3	Numero Historia: 5
Nombre Tarea: Creación de Web Service Rest	
Tipo de Tarea: Programación	Tiempo Estimado: 8 horas
Fecha Inicio: 12/07/2018	Fecha Fin: 13/07/2018
Programador Responsable: Robinson Ruano	
Descripción: Utilizando PHP se implementarán Web Services para obtener la información de la base de datos.	

Fuente: Propia

- **Implementación de funciones principales**

TABLA 3.53 Tareas de historia 6 Tarea 4 Implementación de funciones principales

Tarea	
Numero Tarea: 4	Numero Historia: 6
Nombre Tarea: Implementación de funciones principales	
Tipo de Tarea: Programación	Tiempo Estimado: 12 horas
Fecha Inicio: 13/07/2018	Fecha Fin: 16/07/2018
Programador Responsable: Robinson Ruano	
Descripción: La tarea describe implementar las funciones, llamando a los Web Services creados anteriormente.	

Fuente: Propia

e) Tareas de Historia 7: Realizar Pedidos

TABLA 3.54 Tareas de historia 7

Nro.	Nombre	Tiempo Estimado
1	Diseño de botones para cada opción	8 horas
2	Creación de Web Service Rest	10 horas
3	Implementación de funciones principales	14 horas
	TOTAL	32 horas

Fuente: Propia

- **Diseño de botones para cada opción**

TABLA 3.55 Tareas de historia 7 Tarea 1 Diseño de botones para cada opción

Tarea	
Numero Tarea: 1	Numero Historia: 7
Nombre Tarea: Diseño de botones para cada opción	
Tipo de Tarea: Diseño	Tiempo Estimado: 8 horas
Fecha Inicio: 17/07/2018	Fecha Fin: 17/07/2018
Programador Responsable: Robinson Ruano	
Descripción: Se realizará un diseño que sea simple y atractivo para el cliente. Que cuente con todas las opciones que necesite el usuario.	

Fuente: Propia

- **Creación de Web service Rest**

TABLA 3.56 Tareas de historia 7 Tarea 1 Diseño de botones para cada opción

Tarea	
Numero Tarea: 2	Numero Historia: 6
Nombre Tarea: Creación de Web service Rest	
Tipo de Tarea: Programación	Tiempo Estimado: 10 horas
Fecha Inicio: 18/07/2018	Fecha Fin: 19/07/2018
Programador Responsable: Robinson Ruano	
Descripción: Utilizando PHP se implementarán Web Services para obtener la información de la base de datos	

Fuente: Propia

- **Implementación de funciones principales**

TABLA 3.57 Tareas de historia 7 Tarea 1 Implementación de funciones principales

Tarea	
Numero Tarea: 3	Numero Historia: 6
Nombre Tarea: Implementación de funciones principales	
Tipo de Tarea: Programación	Tiempo Estimado: 14 horas
Fecha Inicio: 19/07/2018	Fecha Fin: 20/07/2018
Programador Responsable: Robinson Ruano	
Descripción: La tarea describe implementar las funciones, llamando a los Web Services creados anteriormente	

Fuente: Propia

3.2. Diseño de la Aplicación

En esta fase se diseña la aplicación a partir de los requerimientos agrupados de las historias de usuarios anteriormente mencionados.

3.2.1. Arquitectura de la Aplicación

Para realizar un software de calidad se ha decidido implementar la arquitectura MVC (Modelo Vista Controlador), con el fin de separar los datos, la lógica de negocios y la vista. A continuación, se presentará de manera rápida cada capa del sistema

- a) **Modelo:** Es la capa en la que se realizan las operaciones con los datos de la aplicación, Además de la comunicación con la base de datos para extracción de la información.
- b) **Controlador:** En esta capa se implementa el código que reacciona dependiendo de las solicitudes realizadas por el usuario, dirigiendo el sistema a la clase o vista necesaria para el óptimo funcionamiento del sistema.
- c) **Vista:** En esta capa se presenta en una interfaz gráfica la información ya procesada anteriormente por el Capa de Modelo.

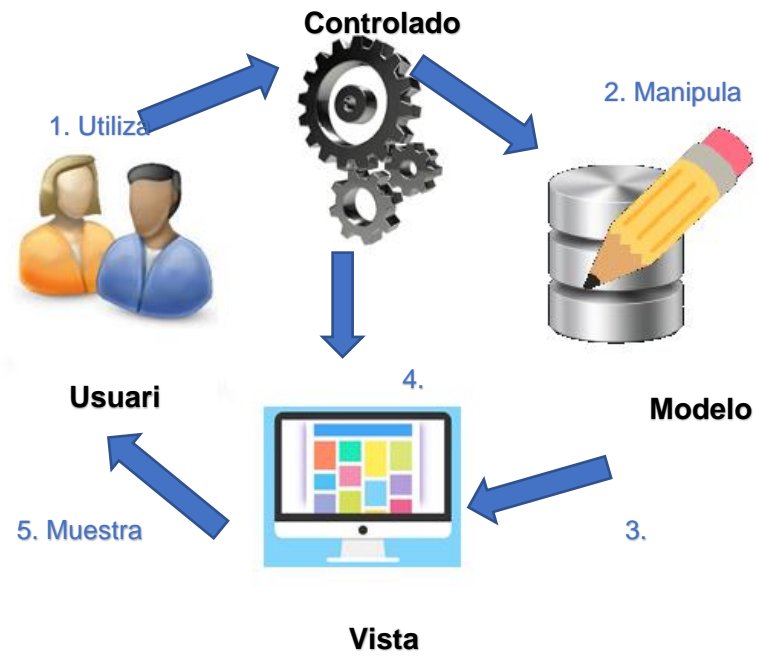


Fig 4. Arquitectura de la Aplicación
Fuente: Propia

3.2.2. Diagrama de Base De Datos

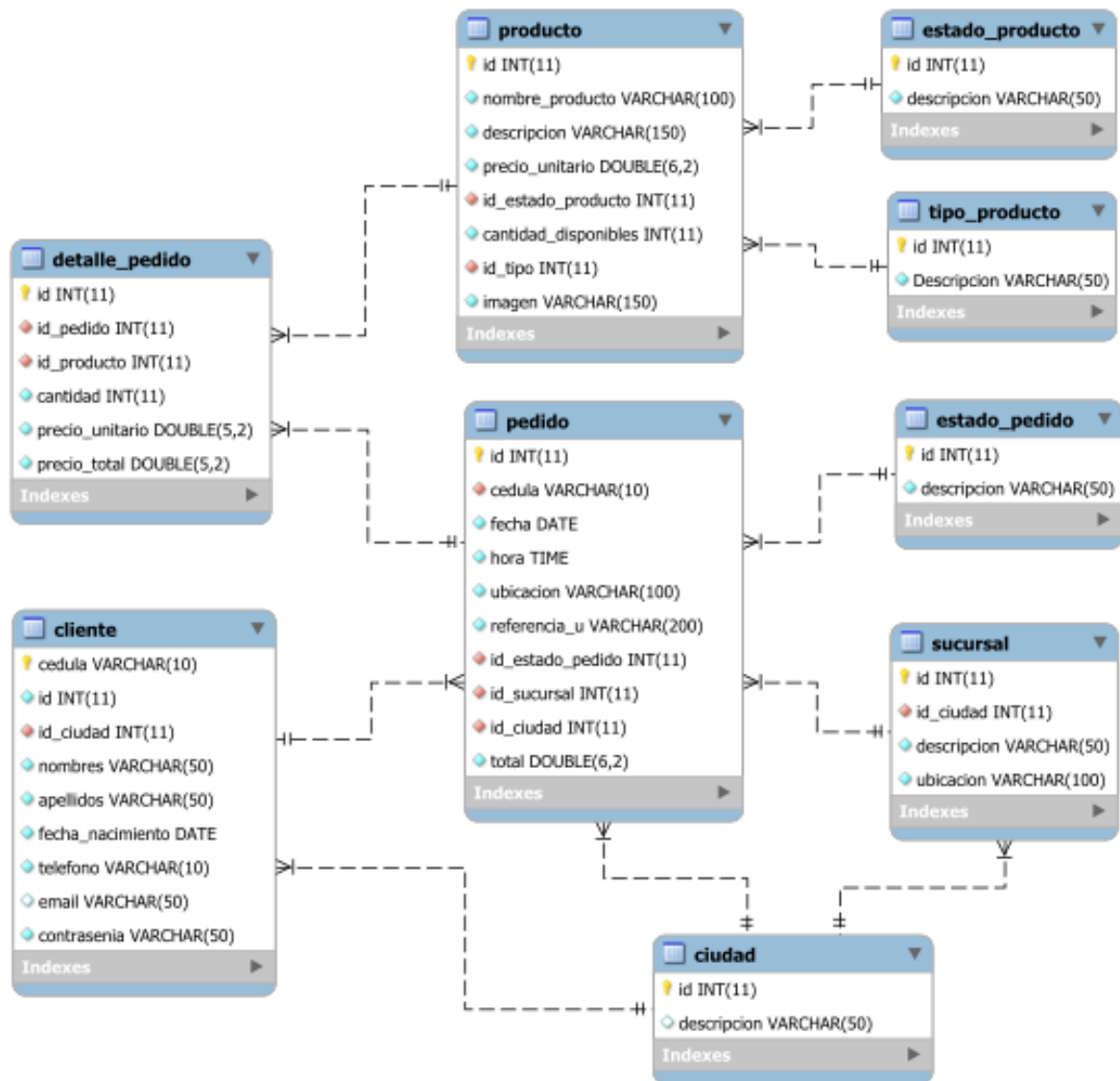


Fig 5. Diagrama de la base de datos

Fuente: Propia

3.2.3. Funcionamiento de la Aplicación

La aplicación contará con dos módulos destinados para dos diferentes tipos de usuarios como: administradores y clientes. Será desarrollada con herramientas Open Source.

Para el módulo de Administradores solo se tendrá acceso desde un navegador, mientras que el módulo de cliente podrá ser accedidos desde un dispositivo móvil que descargue la aplicación sea este con sistema operativo Android o iOS.

3.2.4. Diseño de la Aplicación

Los diferentes módulos contarán con un diseño atractivo e intuitivo, con la finalidad de facilitar que los usuarios realicen sus actividades. La aplicación constará de lo siguiente:

a) Módulo Adminsitrador:

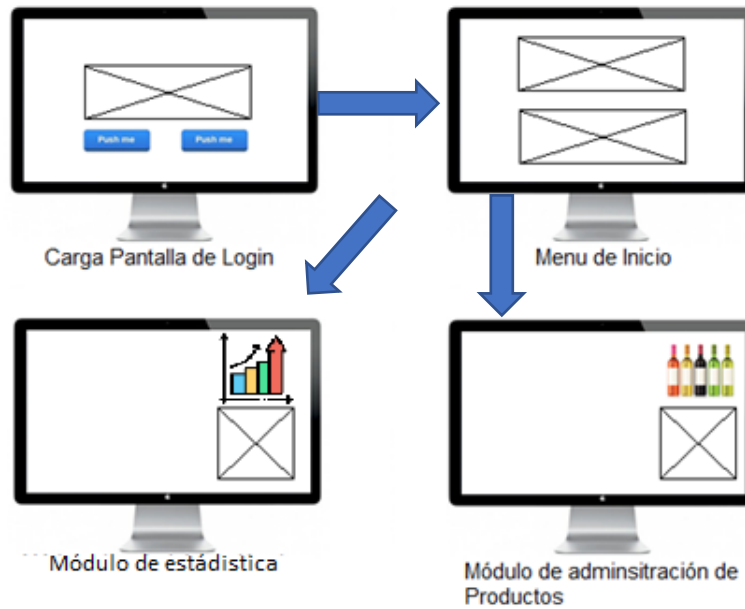


Fig 6. Diagrama del Módulo Administrador

Fuente: Propia

b) Módulo Cliente

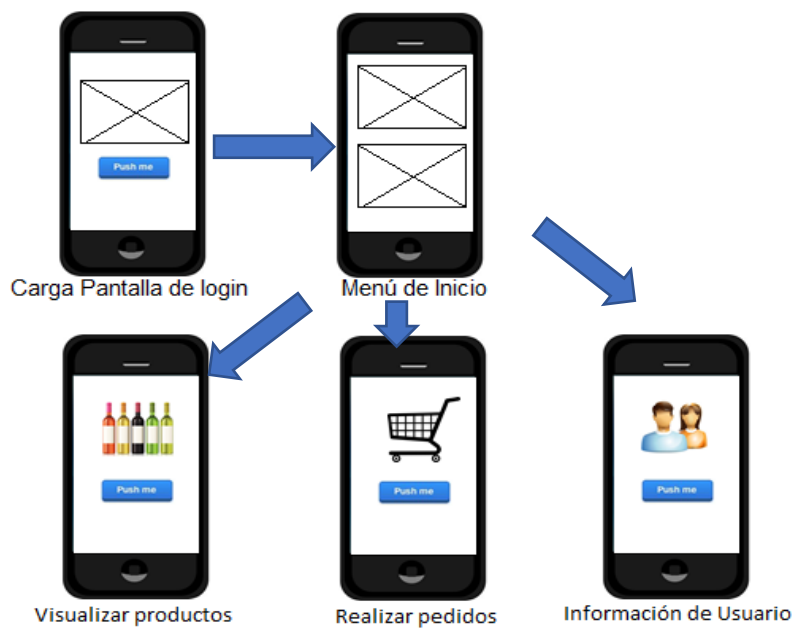


Fig 7. Diagrama del Módulo Cliente

Fuente: Propia

3.3. Pruebas.

3.3.1. Prueba Iteración I

- Prueba de ingresar a la aplicación web

TABLA 3.58 Prueba de ingresar a la aplicación web

Prueba de aceptación	
Caso de prueba: Ingresar a la aplicación web	Opción de prueba: Información General web
Nro. De caso de prueba: 01	Nro. De historia de usuario: 01
Nombre de caso de prueba: Acceso a la aplicación web	
Descripción: Carga e ingresar a la aplicación web.	
Condiciones de ejecución: Acceso a internet Tener instalado un navegador web	
Datos de entrada: Ingresar el usuario y su contraseña	
Pasos de ejecución: Ingresar el Url de la página en el navegador Ingresar los datos solicitados.	
Resultado esperado: La página web carga con normalidad	
Evaluación: Correcto despliegue de la aplicación web.	

Fuente: Propia

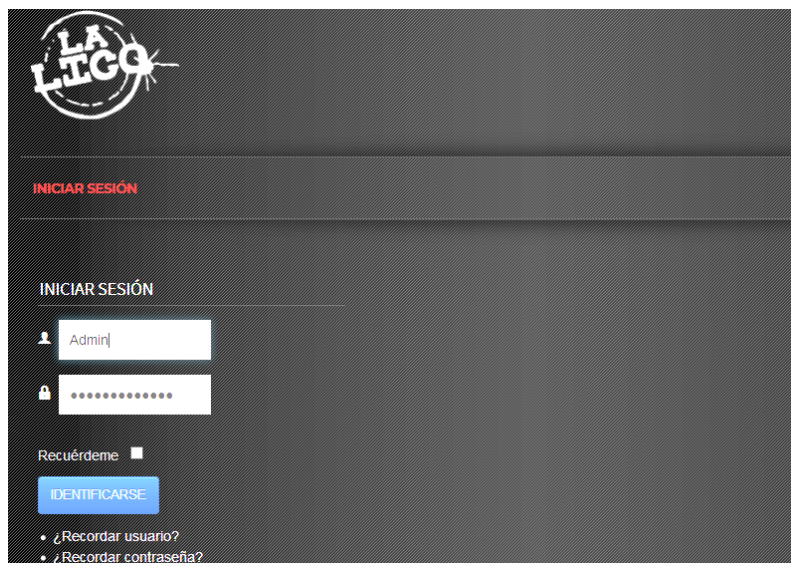


Fig 8. Login del portal web

Fuente: Propia

- **Prueba de visualización de datos en el sistema**

TABLA 3.59 Prueba de visualización de datos en el sistema

Prueba de aceptación	
Caso de prueba: Gestionar datos en el sistema	Opción de prueba: Datos
Nro. De caso de prueba: 02	Nro. De historia de usuario: 02
Nombre de caso de prueba: Gestión del Portal	
Descripción: Visualización de datos que se mostraran en la aplicación.	
Condiciones de ejecución: Ingresar al sistema con usuario y contraseña correctos.	
Datos de entrada: No existen datos de entrada.	
Pasos de ejecución: Ingresar a un módulo desde el menú de inicio.	
Resultado esperado: La aplicación permite visualizar los datos de la base de datos.	
Evaluación: Correcto funcionamiento de la aplicación web.	

Fuente: Propia







Nombre	Descripción	Precio Unitario	Cantidad	Estado	Tipo	Imagen	Editar
Grands	Botella de 600cc	15.00	201	Activado	Whisky		
Paisa Sandia	Botella de 660cc	7.20	120	Activado	Aguardiente		
Corona	Cerveza extra ligera	2.49	200	Activado	Cerveza		

Fig 9. Gestor de productos

Fuente: Propia

- **Prueba de gestión de datos en el sistema**

TABLA 3.60 Prueba de gestión de datos en el sistema

Prueba de aceptación	
Caso de prueba: Gestionar datos en el sistema	Opción de prueba: Datos
Nro. De caso de prueba: 03	Nro. De historia de usuario: 02
Nombre de caso de prueba: Gestión del Portal	
Descripción: Ingreso, edición y actualización de datos.	
Condiciones de ejecución: Ingresar al sistema con usuario y contraseña correctos. Tener permisos de edición.	
Datos de entrada: Ingresar los datos pedidos en cada clasificación del módulo.	
Pasos de ejecución: Ingresar a un módulo desde el menú de inicio. Agregar un nuevo ítem. Editar un ítem.	
Resultado esperado: La aplicación permite ingresar, editar, actualizar y visualizar ítems	
Evaluación: Correcto funcionamiento de la gestión de ítems.	

Fuente: Propia

Fig 10. Ingreso de productos

Fuente: Propia

- Prueba de estadísticas

TABLA 3.61 Prueba de estadísticas

Prueba de aceptación	
Caso de prueba: Visualizar estadísticas generadas por el uso del sistema.	Opción de prueba: Datos
Nro. de caso de prueba: 04	Nro. De historia de usuario: 02
Nombre de caso de prueba: Gestión del Portal	
Descripción: Visualización de gráficos estadísticos.	
Condiciones de ejecución: Ingresar al sistema con usuario y contraseña correctos. Tener transacciones realizadas.	
Datos de entrada: Ninguno.	
Pasos de ejecución: Ingresar a un módulo desde el menú de inicio.	
Resultado esperado: La aplicación permite ver gráficos estadísticos tales como: productos más vendidos, productos menos vendidos, ventas mensuales por año, productos menos despachados.	
Evaluación: Correcta visualización de cada gráfico.	

Fuente: Propia



Fig 11. Módulo de estadísticas

Fuente: Propia

- **Prueba de visualización de pedidos**

TABLA 3.62 Prueba de visualización de pedidos

Prueba de aceptación	
Caso de prueba: Visualización de pedidos	Opción de prueba: Datos
Nro. de caso de prueba: 05	Nro. De historia de usuario: 03
Nombre de caso de prueba: Visualización de pedidos	
Descripción: Visualización de pedidos categorizados según su estado.	
Condiciones de ejecución: Ingresar al sistema con usuario y contraseña correctos. Tener transacciones realizadas.	
Datos de entrada: Ninguno.	
Pasos de ejecución: Ingresar a un módulo desde el menú de inicio.	
Resultado esperado: La aplicación permite ver todos los pedidos realizados en el sistema, separarlos por su estado.	
Evaluación: Correcta visualización de cada pedido realizado en él sistema.	

Fuente: Propia

ID	Cédula	Fecha	Hora	Estado	Total
82	0401709696	2018-10-23	12:16:00	Pendiente	\$198.73
81	0401709696	2018-10-23	00:00:00	Entregado	\$33.25
80	0401709696	2018-10-22	00:00:00	Entregado	\$25.50
79	0401635354	2018-10-21	23:59:02	Entregado	\$14.85
78	0402090864	2018-10-18	00:00:00	Entregado	\$75.00
77	0401709696	2018-10-18	00:00:00	Cancelado	\$30.00
76	0402090864	2018-10-18	00:00:00	Entregado	\$12.45
75	0402090864	2018-10-18	00:00:00	Entregado	\$141.00
74	0401709696	2018-10-16	00:00:00	Cancelado	\$15.00
73	0401709696	2018-10-16	00:00:00	Entregado	\$35.50

Fig 12. Módulo de historial de pedidos

Fuente: Propia

- **Prueba de visualización del detalle de cada pedido**

TABLA 3.63 Prueba de visualización del detalle de cada pedido

Prueba de aceptación	
Caso de prueba: Visualización de pedidos	Opción de prueba: Datos
Nro. de caso de prueba: 06	Nro. De historia de usuario: 03
Nombre de caso de prueba: Visualización del detalle de cada pedido	
Descripción: Visualizar el detalle de cada pedido como: la ubicación desde donde se realizado, el cliente que lo realizo, además de la información detallada de cada producto.	
Condiciones de ejecución: Ingresar al sistema con usuario y contraseña correctos. Tener transacciones realizadas.	
Datos de entrada: Ninguno.	
Pasos de ejecución: Ingresar a un módulo desde el menú de inicio. Dar clic sobre un pedido.	
Resultado esperado: La aplicación permite ver todos los datos necesarios para poder entregar el pedido a su cliente.	
Evaluación: Correcta visualización de los detalles de cada pedido.	

Fuente: Propia



Fig 13. Detalle del pedido

Fuente: Propia

3.3.2. Prueba Iteración II

- Prueba de Acceso a la Aplicación del Cliente

TABLA 3.64 Prueba de Acceso a la Aplicación del Cliente

Prueba de aceptación	
Caso de prueba: Ingresar a la aplicación móvil	Opción de prueba: Información General
Nro. De caso de prueba: 07	Nro. De historia de usuario: 04
Nombre de caso de prueba: Acceso a la Aplicación del Cliente	
Descripción: Carga e ingresar a la aplicación móvil.	
Condiciones de ejecución: Tener instalado la aplicación móvil. Acceso a internet.	
Datos de entrada: Ingresar datos: email y contraseña	
Pasos de ejecución: Ejecutar aplicación. Ingresar con su email y contraseña	
Resultado esperado: La aplicación inicia con normalidad y permite ingresar al menú	
Evaluación: Correcto despliegue de la aplicación.	

Fuente: Propia



Fig 14. Login de la aplicación web

Fuente: Propia

- **Prueba de registro en la aplicación**

TABLA 3.65 Prueba de registro en la aplicación

Prueba de aceptación	
Caso de prueba: Registrarse en la aplicación	Opción de prueba: Información General
Nro. De caso de prueba: 08	Nro. De historia de usuario: 04
Nombre de caso de prueba: Acceso a la Aplicación del Cliente	
Descripción: Ingreso de datos personales para registrarse en la base de datos de la aplicación.	
Condiciones de ejecución: Tener instalado la aplicación móvil. Acceso a internet.	
Datos de entrada: Ingresar datos personales: nombres, apellidos, email, teléfono etc.	
Pasos de ejecución: Ejecutar aplicación. Crear una cuenta en la opción de registrar.	
Resultado esperado: La aplicación inicia con normalidad y permite registrar usuarios.	
Evaluación: Correcto despliegue de la aplicación e ingreso de datos.	

Fuente: Propia

Fig 15. Registro de clientes

Fuente: Propia

- **Prueba de visualización del menú de opciones**

TABLA 3.66 Prueba de visualización del menú de opciones

Prueba de aceptación	
Caso de prueba: Visualización del menú	Opción de prueba: Información General
Nro. De caso de prueba: 09	Nro. De historia de usuario: 05
Nombre de caso de prueba: Visualizar menú de opciones	
Descripción: Visualización del menú de opciones con todas las funciones de la aplicación.	
Condiciones de ejecución: Tener instalado la aplicación móvil. Acceso a internet.	
Datos de entrada: Ninguno.	
Pasos de ejecución: Ejecutar aplicación. Ingresar a la aplicación con usuario y contraseña	
Resultado esperado: La aplicación inicia con normalidad y permite ver todas las funcionalidades que ofrece.	
Evaluación: Correcta visualización de opciones.	

Fuente: Propia

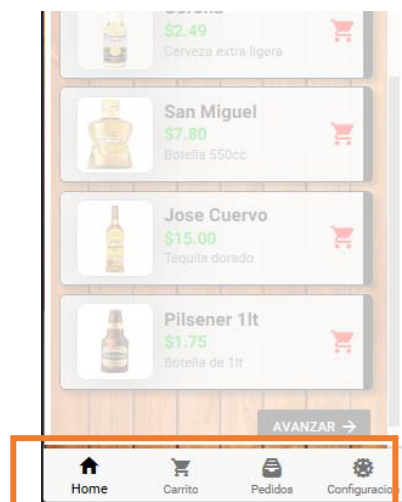


Fig 16. Menú de la aplicación móvil

Fuente: Propia

- **Prueba de visualización de productos y promociones**

TABLA 3.67 Prueba de visualización de productos y promociones

Prueba de aceptación	
Caso de prueba: Visualización del menú	Opción de prueba: Información General
Nro. De caso de prueba: 10	Nro. De historia de usuario: 06
Nombre de caso de prueba: Visualizar productos y promociones	
Descripción: Visualización de todos los productos que están publicados en la aplicación	
Condiciones de ejecución: Tener instalado la aplicación móvil. Acceso a internet.	
Datos de entrada: Ninguno.	
Pasos de ejecución: Ejecutar aplicación. Ingresar a la aplicación con usuario y contraseña	
Resultado esperado: La aplicación permite visualizar todos los productos activos que se encuentran en la base de datos.	
Evaluación: Correcta visualización de productos y promociones.	

Fuente: Propia

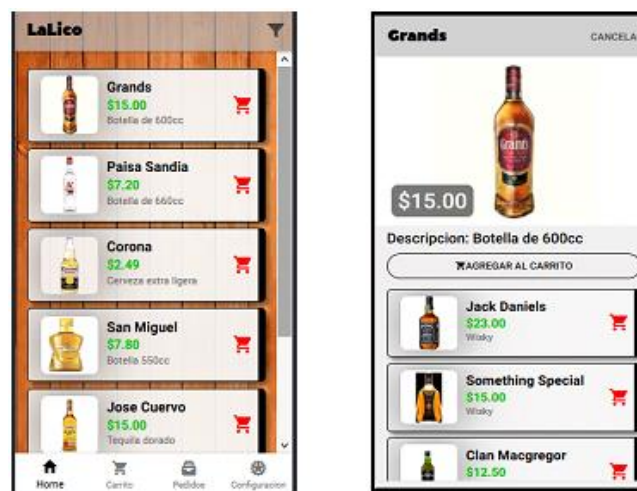


Fig 17. Visualización de productos

Fuente: Propia

- **Prueba de realizar pedidos**

TABLA 3.68 Prueba Agregar al carrito de compras

Prueba de aceptación	
Caso de prueba: Realizar pedidos	Opción de prueba: Datos
Nro. De caso de prueba: 11	Nro. De historia de usuario: 07
Nombre de caso de prueba: Agregar al carrito de compras	
Descripción: Agregar un producto al carrito de comprar.	
Condiciones de ejecución: Tener instalado la aplicación móvil. Acceso a internet.	
Datos de entrada: Ninguno.	
Pasos de ejecución: Seleccionar un producto. Definir la cantidad deseada Clic en la opción de agregar.	
Resultado esperado: La aplicación permite agregar los productos deseados al carrito de compras.	
Evaluación: Correcta visualización de opciones.	

Fuente: Propia

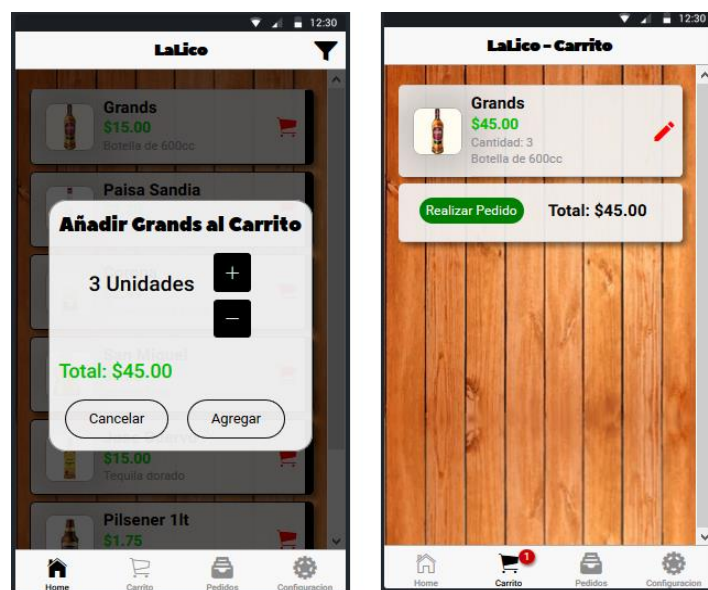


Fig 18. Agregar al carrito

Fuente: Propia

- **Prueba de realizar pedido**

TABLA 3.69 Prueba realizar pedidos

Prueba de aceptación	
Caso de prueba: Realizar pedido	Opción de prueba: Datos
Nro. De caso de prueba: 12	Nro. De historia de usuario: 07
Nombre de caso de prueba: Confirmar pedido.	
Descripción: Confirmar el pedido para que el administrador pueda verlo.	
Condiciones de ejecución: Tener instalado la aplicación móvil. Acceso a internet. Tener agregado productos al carrito.	
Datos de entrada: Ingresar la referencia de la ubicación.	
Pasos de ejecución: Clic en realizar pedido. Agregar referencia de ubicación. Clic en confirmar pedido.	
Resultado esperado: La aplicación permite realizar el pedido y se guarda en la base de datos.	
Evaluación: El pedido se realiza con éxito.	

Fuente: Propia

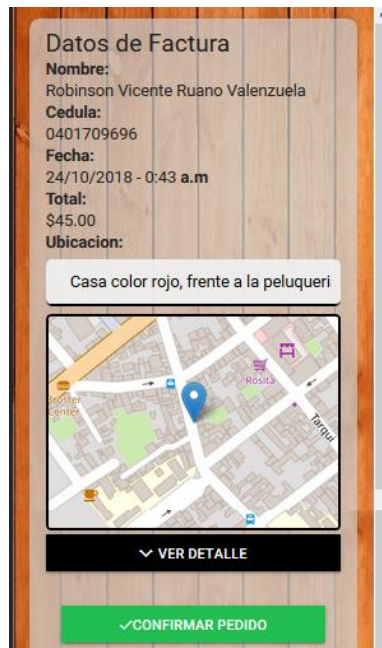


Fig 19. Detalle del pedido

Fuente: Propia

CONCLUSIONES

- La investigación de los dos frameworks da como resultado que ambas herramientas son muy poderosas para el desarrollo de aplicaciones móviles multi-plataforma, dando como resultado aplicaciones de calidad con las mismas funciones y características que tiene una aplicación creada con su propio lenguaje nativo.
- La normativa ISO-9126 ayudo a evaluar la calidad de los frameworks analizados en la presente tesis, aplicando las métricas de evaluación como: portabilidad, funcionalidad, fiabilidad, usabilidad y eficiencia. La métrica de mantenibilidad no fue tomada en cuenta debido a que no se puede obtener acceso al código fuente de los frameworks y calificar su mantenibilidad.
- La metodología de desarrollo ágil Extreme Programming (XP), permite que los desarrolladores puedan desarrollar software de calidad en un periodo corto de tiempo, debido a que el cliente también está involucrado en el proyecto se obtiene una retroalimentación en tiempo real, mejorando así la comunicación entre en programador y el cliente.
- El framework que obtuvo mejor puntuación fue Ionic que supero a React Native por 0,70 puntos convirtiéndose así en el software ganador, así se define como el framework mejor equipado para la realización de aplicaciones móviles multiplataforma. Siendo así la herramienta usada en el desarrollo de la aplicación “LalicoApp” para la empresa “La Lico”, dando como resultado una aplicación rápida, confiable y de calidad, con funciones iguales a las de una aplicación nativa.

RECOMENDACIONES

- Para realizar un proyecto de desarrollo se recomienda primero investigar las posibles herramientas a usar, evaluando sus características y funciones.
- Se debe aplicar normas internacionales que den ciertos parámetros a seguir o cumplir para obtener un producto de calidad, que cumpla con las expectativas de los clientes.
- Fomentar la aplicación de metodologías de programación que permiten optimizar el desarrollo de una aplicación, siguiendo el ciclo de vida de software. Sobre todo, el uso de metodologías ágiles que se centran en el desarrollo del sistema y dejan de lado la documentación del mismo.
- Aprender varios frameworks para el desarrollo de aplicaciones y utilizar la mejor opción, dependiendo de las necesidades del cliente.
- Es necesario revisar las actualizaciones del framework Ionic debido a que siempre está integrando funciones nuevas.
- Al cambiar de versión del framework se recomienda realizar la migración del proyecto usando la documentación que ofrece el framework.

REFERENCIAS

- Apache Cordova. (27 de 3 de 2018). *Cordova Introducción*. Obtenido de Cordova Introducción: <https://cordova.apache.org/docs/en/latest/guide/overview/index.html>
- Apple Inc. (30 de 11 de 2017). *developer apple Inc*. Obtenido de Apple Inc Web Site: https://developer.apple.com/library/content/documentation/MacOSX/Conceptual/OSX_Technology_Overview/
- Boduch, A. (2017). *React and React Native*. Birmingham: Packt Publishing .
- Contreras, M. (2016). *Desarrollo de aplicaciones web multiplataforma*. Madrid: Ministerio de Educación de España.
- Cuello, J., & Vittone, J. (2013). *Diseñando apps para móviles*. Buenos Aires: TugaMovil.
- Domínguez, F. (2014). *Programación multimedia y dispositivos móviles*. Santacruz: RA-MA Editorial.
- Dpto de Ciencia de la Computación e Inteligencia Artificial. (13 de 11 de 2018). *Desarrollo de Aplicaciones para Android*. Obtenido de <http://www.jtech.ua.es/>: <http://www.jtech.ua.es/apuntes/ajdm2010/sesiones/sesion09-apuntes.html#Android>
- Durán, D. (2015). *Gestión de la calidad de productos editoriales multimedia*. Antequera: IC Editorial.
- Esteban Vázquez-Cano, M. L. (2015). *Dispositivos digitales móviles en educación: el aprendizaje ubicuo*. Madrid: Narcea Ediciones.
- Extreme Programming. (3 de 4 de 2018). *La simplicidad es la clave*. Obtenido de La simplicidad es la clave: <http://www.extremeprogramming.org/rules/simple.html>
- Facebook. (23 de Enero de 2018). *facebook.github.io*. Obtenido de facebook.github.io: <https://facebook.github.io/flux/docs/in-depth-overview.html#content>
- Fernández, I. (2015). *Construcción de una escala de actitudes tipo Likert*. Madrid: Instituto nacional de higiene y trabajo.
- Flux Facebook. (20 de 2 de 2018). *Flux React*. Obtenido de Flux React: <https://facebook.github.io/flux/docs/in-depth-overview.html#content>

- Gallego, A. (2017). *Manual de introducción a Ionic*. Alcantara.
- García, C., Gómez, D., Rubén, S., & Molina, L. (2017). *Introducción a la informática básica*. Madrid: Universidad Nacional de Educación a Distancia.
- Gironés, T. (2014). *EL gran libro del Android*. Mexico D.F.: Alfaomega.
- Gómez, J. (2016). *dirección y gestión de proyectos de tecnologías de la información en la empresa*. Madrid: FC Editorial.
- Gordon, C., Rezzadeh, K., & Li, A. (2015). Digital mobile technology facilitates HIPAA-sensitive perioperative messaging, improves physician-patient communication, and streamlines patient care. *BioMed Central*, N/A.
- Granados, R. (2014). *Despliegue y puesta en funcionamiento de componentes software: UF1291*. Málaga: IC Editorial.
- Hernández, R., Fernández, C., & Baptista, P. (2014). *Metodología de investigación*. Mexico D.F.: McGRAW-HILL.
- Ingavélez, P., Hilera, J., & Timbi, C. (2016). *Ática 2016. tecnología y accesibilidad*. Alcalá: Universidad de Alcalá.
- Ionic. (25 de 3 de 2018). *What is Ionic*. Obtenido de What is Ionic: <https://ionicframework.com/what-is-ionic>
- Laínez, J. (2014). *Desarrollo de Software ÁGIL: Extreme Programming y Scrum*. Createspace Independent Pub.
- Microsoft. (27 de 9 de 2018). *msdn*. Obtenido de <https://blogs.msdn.microsoft.com/typescript/2018/09/27/announcing-typescript-3-1/>
- Montiel, A. (2017). *El mobile marketing y las apps: cómo crear apps e idear estrategias de mobile marketing*. Barcelona: UOC.
- Ravulavaru, A. (2015). *Learning Ionic*. MUMBAI: Pactk Publishing.
- React Facebook. (21 de 1 de 2018). *React*. Obtenido de React: <https://reactjs.org/docs/introducing-jsx.html>
- React Native. (14 de 11 de 2017). *React Native*. Obtenido de React Native: <https://facebook.github.io/react-native/>

React Native. (26 de 1 de 2018). *Data Flow React Native*. Obtenido de Data Flow eact Native:
<https://facebook.github.io/react-native/docs/communication-ios>

Sacristán, C., & Fernández, D. (2012). *Programación en Android*. Mdrid: Aula mentor.

Sanz, R., & López, R. (2014). *Introducción a la movilidad: 4G/LTE y el desarrollo de aplicaciones Android*. Madrid: Dextra Editorial.

UNESCO. (2013). *Policy Guidelines for Mobile Learning*. Paris: UNESCO.

w3schools. (20 de 01 de 2018). *Data Refsnes*. Obtenido de Data Refsnes:
https://www.w3schools.com/js/js_htmlDOM.asp

Wargo, J. (2015). *Apache Cordova API CookBook*. Westford: Addison-Wesley.

ANEXOS

Anexo A: Capturas de pantalla de la aplicación desarrollada con React Native

