PONTIFICIA UNIVERSIDAD CATOLICA DEL ECUADOR

FACULTAD DE INGENIERIA

INGENIERÍA DE SISTEMAS Y COMPUTACIÓN

DISERTACIÓN PREVIA A LA OBTENCIÓN DEL TÍTULO DE INGENIERO DE SISTEMAS Y COMPUTACIÓN

ANÁLISIS COMPARATIVO DE FRAMEWORKS FRONTEND PARA APLICACIONES MÓVILES A TRAVÉS DE UN APLICATIVO WEB CON INTELIGENCIA ARTIFICIAL.

AUTOR: ADRIANA MISHELL PAZMINO VILLAMARÍN

DIRECTOR: Mgtr. CHARLES ESCOBAR

QUITO DM, 2022

DEDICATORIA

El presente trabajo de titulación está dedicado para mis padres y hermana quienes han sido un pilar fundamental en el transcurso de mi vida y ha sido parte de mi desarrollo profesional.

AGRADECIMIENTO

Agradezco a Dios por permitirme tener vida y poder gozar de la compañía de mis seres queridos.

Agradezco a mis padres por forjar a una mujer de valentía, que se plantea metas y las cumple con determinación, y sobre todo por ser un apoyo incondicional durante toda la vida.

A mi hermana por ser un ejemplo de superación y cariño, gracias por siempre ser mi cómplice de aventuras y aconsejarme con tu experiencia.

A mi mejor amigo Leonardo quien ha sido apoyo incondicional en los buenos y malos momentos, gracias por estar siempre al pendiente y brindarme una amistad sincera.

Al Ingeniero Charles Escobar, mi gratitud por la apertura y bondad que proyecta, por los conocimientos impartidos que han servido de base e inspiración para el desarrollo de esta investigación.

Adriana

Tabla de contenido

Índice de Figuras

Índice de Tablas

Resumen

El presente trabajo ha sido desarrollado con el objetivo de proveer una solución que permita a los desarrolladores de aplicaciones móviles elegir un entorno que se adapte a los requerimientos de un proyecto, ya que al disponer de varios softwares resulta complejo decidir cuál de estos frameworks se debe utilizar. Para esto se analizó los entornos más populares tales como: Flutter, React Native, Ionic, Xamarín y Cordova. Luego de identificar las características que presenta cada framework, se realizó una comparativa que indica las condiciones para definir las ventajas relevantes. Para sustentar el análisis previamente realizado se desarrolló un sitio web que funciona mediante un proceso de inteligencia artificial, que predice el nombre del entorno que mejor se adapta a las necesidades de un proyecto a implementarse.

Se realizó una encuesta a una población que se dedica a programar aplicaciones móviles, con la finalidad de obtener las preferencias y gustos en diferentes ámbitos de los framework. A partir de esta recolección de datos se generó un dataset que contiene variables de entrada (características más representativas del entorno) y variables de salida (entornos analizados). Para el procesamiento de los datos obtenidos se evaluó los modelos de clasificación y como resultado se determinó que el algoritmo de redes neuronales presentó mejores resultados en cuanto a eficiencia. Finalmente, a través de un sitio web se visualizarán los resultados obtenidos que van acorde a los objetivos de esta investigación.

Palabras clave: Framework, Inteligencia Artificial, Aplicaciones móviles, desarrolladores.

Abstract

This work has been developed with the aim of providing a solution that allows mobile application developers to choose an environment that suits the requirements of a project, since having several softwares is complex to decide which of these frameworks should be used. For this purpose, the most popular frameworks such as: Flutter, React Native, Ionic, Xamarin and Cordova were analyzed. After identifying the characteristics of each framework, a comparative analysis was carried out, indicating the conditions to define the relevant advantages. To support the analysis previously carried out, a website was developed that works through an artificial intelligence process, which predicts the name of the framework that best suits the needs of a project to be implemented.

A survey was made to a population dedicated to programming mobile applications, in order to obtain the preferences and tastes in different areas of the framework. From this data collection, a dataset was generated containing input variables (most representative characteristics of the environment) and output variables (environments analyzed). For the processing of the data obtained, the classification models were evaluated and as a result it was determined that the neural network algorithm presented better results in terms of efficiency. Finally, a web site will be used to visualize the results obtained in accordance with the objectives of this research.

Palabras clave: Framework, Artificial Intelligence, Mobile applications, developers.

1. CAPÍTULO I: INTRODUCCIÓN

1.1 Generalidades

El impacto de la era tecnológica cada vez se hace más notorio dentro de la sociedad y la aparición de nuevas herramientas se han vuelto necesarias al momento de realizar un diseño e implementación de un software, ya sea móvil o web. Por tal razón es importante conocer cuáles son los nuevos frameworks orientados al diseño de aplicaciones. Pero al mismo ritmo que surgen estas opciones, se vuelve necesario identificar como estos entornos interactúan y en función de qué parámetros se puede establecer las características que diferencian a cada uno para su utilización.

"Además, se ha identificado que existe una tendencia importante en la utilización de aplicaciones móviles, debido a que los usuarios tienen la necesidad de mantenerse conectados a través de dispositivos inteligentes y pueden acceder a sus aplicaciones favoritas desde cualquier parte del mundo, por esta razón los desarrolladores buscan mantenerse a la vanguardia de herramientas que facilitan la implementación y mantienen la usabilidad en los usuarios" (Cárdenas Villavicencio et al., 2021).

Según Yaguapaz (2018) "la creación de nuevas herramientas ha proporcionado una ayuda importante para los desarrolladores, ya que se ha disminuido el tiempo de implementación, disminuyendo los costos planificados para estas aplicaciones, y la facilidad que brindan los distintos frameworks se ha incrementado su utilización. En la actualidad se encuentra este tipo de software en casi cualquier campo, y se los utiliza para ofrecer servicios o mejorar procesos".

Un framework frontend es un entorno o ambiente que se utiliza para la implementación de aplicaciones, su uso se enfoca en la adecuada interacción entre el cliente y el aplicativo. Estos marcos de trabajo brindan una serie de herramientas a las personas que se dedican a desarrollar, permitiéndoles así dar a sus aplicativos un aspecto innovador, en donde se optimiza la interfaz de las aplicaciones y la experiencia de los usuarios.

Este trabajo se enfoca en identificar las herramientas de desarrollo frontend existentes estableciendo: el entorno adecuado de trabajo, tasa de aprendizaje, costo, utilidad, compatibilidad con sistemas, escalabilidad y adaptabilidad a los requerimientos de diseño. En donde será necesario la recolección de información para el estudio de variables

y escenarios, y a través de un algoritmo de aprendizaje supervisado, que se programará en un software, permitirá clasificar la información, para identificar y mostrar en un sitio web el framework que cumpla con los requerimientos que el proyecto demande.

Con este estudio se espera disminuir el tiempo de planificación en los proyectos que se encuentran en fase inicial, también ayuda a solventar la falta de experiencia en el uso de estas herramientas. Es así como a través de los resultados obtenidos se conocerá que entornos se adaptan más a los requerimientos de empresas o personas que se dedican al campo del diseño y experiencia del usuario. Además, con este análisis se tendrá una visión más amplia sobre un posible reajuste, dependiendo de nuevos requerimientos, sin necesidad de modificar la estructura principal por completo.

1.2 Justificación

A través de los años los requerimientos al momento de implementar un producto de software han crecido en tamaño y en complejidad, y las herramientas de desarrollo han ido evolucionando con el fin de cumplir con las necesidades de los ingenieros de software. Actualmente se cuenta con una amplia gama de frameworks el cual se define como "un conjunto estandarizado de conceptos, prácticas y criterios" que se unen con el fin de estandarizar el marco de trabajo para un proyecto e IDE's (Sultan, 2017).

A medida que aumenta la utilización de estas herramientas, se exige un mayor nivel de especialización, que en ciertas ocasiones se convierte en una limitante para los programadores principiantes que no cuentan con la experiencia para la implementación de un software, y por otro lado es una oportunidad ya que se cuenta con una gama más amplia de opciones, que mediante un análisis de datos permita aprovechar los recursos disponibles y sirvan de apoyo para la decisión y selección de las alternativas existentes. Para realizar este análisis se debe recopilar, almacenar y actualizar la información acerca de las diferentes opciones disponibles, lo que implica un proceso de limpieza, filtrado y transformación (ETL), esto con la finalidad de ayudar al programador a tomar una decisión más acertada y en el menor tiempo posible al buscar el entorno ideal para su proyecto.

Otro aspecto importante es la utilización de inteligencia artificial la cual permite discernir de manera más analítica la información, a través de un aprendizaje de máquina supervisado, el cuál evalúa parámetros de entrada, los depura mediante un tipo de algoritmo, y finalmente se obtiene un resultado, que respalda la toma de decisión, para la selección del entorno para el desarrollo de la aplicación móvil.

1.3. Problematización

Aunque en el mercado existe una gran variedad de herramientas para el desarrollo de aplicaciones móviles no se conoce con claridad cuál de estas se debe usar para la implementación de una aplicación. Además, no se cuenta con un estudio que analice los tipos de framework frontend móviles, dependiendo de sus características más importantes y en función de los requerimientos específicos en los que un proyecto se debería desarrollar.

Asimismo, no se ha recopilado información acerca de los tipos de framework frontend para aplicaciones móviles, que pueda servir de base para analizar entornos que permitan desarrollar aplicativos.

En lo que tiene que ver a si un framework es más eficiente que otro y en qué casos es más utilizado, no se han identificado que parámetros pueden servir de comparativa entre ellos ni se los ha cotejado información

Aunque se espera contar con un dataset de preferencia de los desarrolladores por entornos de programación para aplicaciones móviles, no se conoce que algoritmo de aprendizaje automático podría utilizarse para predecir de acuerdo con las variables de entrada que entorno de programación se podría usar.

Tampoco se cuenta con el diseño e implementación de un sitio web para el ingreso de información de las características de entrada y la visualización de los resultados.

1.4 Objetivos

1.4.1 Objetivo General

Analizar los tipos de framework frontend móviles dependiendo de sus características más importantes y en función de los requerimientos específicos en los que se usan.

1.4.2 Objetivos Específicos

- Recopilar información acerca de los tipos de framework frontend para aplicaciones móviles.
- Identificar los parámetros para determinar porqué un framework es más eficiente que otro y en qué casos es más utilizado.
- Comparar los entornos de desarrollo que se escogieron.
- Analizar qué tipo de algoritmo de aprendizaje automático se utilizará para el correcto funcionamiento de la página.
- Diseñar e Implementar un sitio web para el ingreso de información y la visualización de los resultados.

1.5 Alcance

En este trabajo se realizará un análisis comparativo de frameworks móviles con el cual se pretende utilizar un modelo de inteligencia artificial que permita predecir que entorno se adapta mejor a los requisitos que tiene un proyecto y se pueda visualizar los resultados obtenidos en un sitio web.

Para esto se realizará una encuesta a una población que se encuentra desarrollando aplicativos móviles, misma que servirá para la obtención de datos que serán analizados y ponderados. Esta información será introducida a un modelo de inteligencia artificial, mismo que mediante procesos matemáticos y estadísticos analizará patrones que contienen los datos. En el modelo de inteligencia artificial se utilizará algoritmos de clasificación, estos se analizarán para determinar cuál de estos se adapta mejor al análisis que se desarrolla.

El sitio web se utilizará para capturar los distintos parámetros de análisis como: lenguaje de programación, presupuesto que requiere el proyecto, importancia del tiempo que el framework se encuentra en el mercado, tiempo de desarrollo, campos de aplicación, utilización de herramientas extras para el despliegue de una aplicación, nivel de experiencia, preferencias al implementar, sistema operativo, importancia de la guía de soporte. Para el almacenamiento se utilizará una base de datos no relacional que permita procesar los datos de manera más flexible.

En el funcionamiento del sitió se presentará la información de los entornos y se colocarán preguntas que permitan evaluar los requerimientos de un proyecto. Al seleccionar las respuestas a cada pregunta, se presentará una interfaz con el nombre del entorno que ha sido analizado por el modelo entrenado con inteligencia artificial.

1.6 Metodología

En el presente trabajo se pretende realizar una investigación aplicativa y cualitativa en la cual se va a inferir información a través de un análisis para proporcionar un resultado que sea confiable para el usuario. Para el desarrollo se utilizará la metodología ágil denominada Programación Extrema (XP), en la **Tabla *1*** se describen cada una de las etapas y las actividades en cada una de ellas, esta metodología permite realizar un seguimiento continuo de los avances, aportando flexibilidad y retroalimentación en el proceso de implementación.

Tabla 1 *Aplicación de la metodología Extreme Programming (XP)*

Etapas	Actividades
Planificación	Identificación del problema Definición de temas a investigar Definir reuniones durante el desarrollo de la investigación. Seguimiento de entrega.
Diseño	Diseño de base de datos Diseño de interfaz
Desarrollo	Utilización de estándares Programación Recodificación
Pruebas	Corrección de errores Pruebas de aceptación

Los pasos que se tomará de manera general son los siguientes:

- Se iniciará con una investigación sobre los tipos de entornos de desarrollo para aplicativos móviles que se está utilizando en el mercado.
- Posteriormente se establecerá las cuantificaciones que proporcionarán la importancia de los frameworks para su análisis.
- Luego se procederá a recopilar la información a través de blogs e investigaciones anteriormente realizadas.
- Después de seleccionar la información necesaria, se procesará, comparará y a través de un análisis de datos cuantitativo se determinará en que ocasiones un framework es más necesario que otro.
- Luego de identificar la información y definir patrones encontrados, estos serán almacenados en una base de datos para las entradas del algoritmo de aprendizaje autónomo.
- Finalmente se diseñará e implementará una página web para la visualización de los resultados analizados.

2. CAPÍTULO II: FUNDAMENTOS TEÓRICOS

2.1. Antecedentes

"El incesante mundo de la tecnología dedicada a la telefonía móvil ha propuesto que los nuevos y mejorados dispositivos que utilizan IOT abran un sinfín de posibilidades, que han revolucionado los hábitos de conectividad entre los usuarios. Se ha identificado que cerca del 75% del mercado utiliza dispositivos con el sistema operativo Android, el 23% de dispositivos iOS y un 2% para otros sistemas" (Gutiérrez, 2019).

Los dispositivos que ocupan esta variedad de sistemas operativos lo realizan a través de interfaces de entrada y salida generando así grandes volúmenes de información, estos deben tener una interacción con las aplicaciones que se desarrollan para mejorar la experiencia con el usuario, la computación ha proporcionado modernas herramientas para la implementación de interfaces como son los entornos de desarrollo (Gutiérrez, 2019).

Según Espinoza (2020) menciona que poder elegir un entorno de desarrollo es preciso para el éxito de un proyecto, ya que permiten la creación de aplicaciones robustas y facilitan la programación en capas, en donde se pueden realizar modificaciones en cierto nivel sin la necesidad de reestructurar todo el código.

El acceso a la información según Topón (2020) expresa que cada vez es más factible y con mayor inmediatez, esto gracias a los dispositivos móviles que han facilitado la conectividad y portabilidad, además, que existe una gran variedad en el mercado, por ello las aplicaciones móviles se han vuelto más relevantes y su desarrollo ha tomado mayor fuerza . El funcionamiento de cada dispositivo móvil es diferente y por lo tanto es esencial conocer las características para definir con qué tipo de entorno es más eficiente el desarrollo de una aplicación, considerando aspectos como: adaptabilidad, rendimiento y funcionalidad de estos dispositivos.

Con respecto a los tipos de frameworks Yaguapaz (2018) indica que se han desarrollado desde hace algún tiempo y que utilizan varios lenguajes de programación, esto con el fin de disminuir el tiempo en el desarrollo y el costo de planificación, proponiendo la reutilización de código. Además, los entornos de desarrollo permiten que las aplicaciones implementadas se adapten y ejecuten en el sistema operativo de manera efectiva, brindando confianza a los proyectos.

2.2. Fundamentos Teóricos

2.2.1. Análisis de Datos

El análisis de datos ha sido un avance significativo en la utilización de la información y su estudio ha ido en aumento para la presentación de resultados, en la actualidad la información puede ser procesada para la obtención de análisis estadísticos e informes, que reflejan el estado actual de varias temáticas.

En el desarrollo de las empresas se ha planteado nuevas estrategias, para conocer el estado actual en que se encuentran, esto se consigue por medio del procesamiento de la información, en donde la predicción y análisis de data contribuyen a una mejora del desempeño de la empresas en este ámbito, y la implementación de nuevas soluciones innovadoras son apoyadas por el análisis de información (Zatarain, 2019).

Un término que se ha instaurado en los últimos años es el Big Data y se lo define como una gran cantidad de datos que va en crecimiento, este por su naturaleza compleja necesita de una tecnología avanzada y algoritmos eficientes que proporcionen resultados confiables al analizar dicha data (Oussous et al., 2018). Además, entre las características principales del big data se tiene: el volumen, la velocidad y la variedad; estos son factores determinantes para que la data pueda ser procesada y se obtenga una respuesta para una toma de decisión adecuada

El análisis de datos en un futuro se prevé que será introducida en varios campos de la sociedad, haciendo de esta una herramienta que aporte en el desarrollo de soluciones innovadoras, así como también al crecimiento económico en donde sea implementada.

2.2.2. Inteligencia Artificial

La Inteligencia Artificial (IA) es un aspecto que ha tenido un despunte en la actualidad, en la mayoría esta se la encuentra en dispositivos tecnológicos y gracias al crecimiento de redes y aplicaciones han facilitado la vida cotidiana del ser humano. Según Gutiérrez, (2019) define a IA como una rama de la informática, que a través de la programación diseña o computariza una tarea específica. El objetivo de crear una máquina de aprendizaje es simular comportamientos inteligentes que sean similares al de un humano.

En consecuencia, se ha aplicado IA en varios campos de la informática para la eficiencia de procesos, entre las aplicaciones destacas se tiene: motores de recomendación, sistemas de reconocimiento, minería de datos, sistemas de control automático (Gutiérrez, 2019).

Es importante tomar en cuenta que la IA ha desarrollado varios modelos de aprendizaje de máquina como es el aprendizaje automático, también denominado Machine Learning, el objetivo de este es descubrir conocimiento sin ser programado para poder tomar decisiones inteligentes, un ejemplo donde se puede identificar este algoritmo es en la predicción y sugerencias como lo utilizan Facebook y Google para identificar las preferencias de los usuarios.

En la **Figura *1*** se visualizan los tipos de aprendizaje automático que se subdivide en tres estos son: aprendizaje supervisado, aprendizaje no supervisado y aprendizaje de refuerzo (Rouhiainen, 2018).

Figura 1 *Tipos de Aprendizaje Automático*

Fuente:

El aprendizaje supervisado se desarrolla a partir de data que ya se ha recolectado previamente, a partir de esta recolección se indica como la nueva información que se introducirá debe ser clasificada. Por otro lado, el aprendizaje no supervisado no utiliza la recolección de datos, este se diferencia por desarrollar una capacidad para organizar la información de manera autónoma. Finalmente, el aprendizaje de refuerzo se enfoca en la experiencia, en donde se introduce una entrada cada vez que este actúa de manera adecuada a los datos (Rouhiainen, 2018).

Adicionalmente existe otro tipo de algoritmo que es el de aprendizaje profundo también denominado Deep Learning, este se utiliza para problemas avanzados que implican un volumen extenso de datos, utiliza redes neuronales para tipificar un conjunto de datos y así encontrar patrones y relaciones complejas en la información (Rouhiainen, 2018).

2.2.3. Sistema Recomendador

Un sistema recomendador según Valdiviezo (2019) indica que, se han diseñado con el fin de dar una recomendación a través de preferencias y decisiones de los usuarios. Estos utilizan algunas técnicas de filtrado como son:

2.2.3.1. Filtrados colaborativos

Este tipo de filtro se centra en calcular las recomendaciones a partir de votaciones que una comunidad de usuarios ha realizado sobre varios parámetros.

2.2.3.2. Filtrado basado en contenidos

Se divide en dos sub-filtros: el primero se basa en memoria que puede arrojar un resultado inexacto, pero tiende a poseer una respuesta que puede ser analizada; el segundo se basa en modelos, que lo hace más exacto, pero tiene una mayor complejidad para entender el resultado.

2.3. Marco de desarrollo

La informática se relaciona con varios campos, que al ensamblarse en uno solo se apoyan de una interfaz, que se adapte a las preferencias y experiencias de los usuarios. El desarrollo de estos sistemas interactivos se basa en relacionar al usuario final con la funcionalidad de un sistema o aplicación. Sin un correcto desarrollo existe la posibilidad de que aquellos sistemas complejos y robustos fracasen. Es un campo desafiante de manera que deben ser conservadores sin tener en cuenta el tipo de plataforma en la cual funcionan o las especificaciones de los dispositivos (Ruiz et al., 2018).

Un marco de trabajo es una estructura que proporciona organización, control y reutilización de código que se elabora, evitando crear código repetitivo durante el desarrollo; también minimiza posibles errores, generando una lista con el detalle de cada uno de ellos, esto con el fin que el programador pueda identificarlos con mayor facilidad

Existen ventajas significativas en estos entornos debido a que al automatizar varios procesos y facilitar la programación, se puede realizar procesos más complejos que dependerán también del lenguaje de programación que se desea utilizar para el desarrollo, de esta manera se optimiza el tiempo de programación y disminuye el tiempo de respuesta (Zuñiga, 2020).

Según Valdivia (2016) considera que un marco de trabajo “no es un patón sistémico, pero si proporciona una colección de patrones de diseño y clases que operan conjuntamente para la resolución de problemas determinados”. Debido a esto existe una variedad de frameworks con diferentes lenguajes de programación como: Javascript, Java, Php, Python, etc.

“Los marcos de software constan de puntos congelados y puntos calientes. Los puntos congelados definen la arquitectura general de un sistema de software, es decir, sus componentes básicos y las relaciones entre ellos”. Éstos se mantienen inalterados en cualquier instanciación en donde se desee aplicar. Los puntos calientes simbolizan a varias partes en las que los programadores que utilizan un framework colocan su código para una funcionalidad más personalizada en el desarrollo de un proyecto (Valdivia, 2016).

2.4. Librerías

Existe otro tipo de concepto que tiene similitud a un marco de trabajo y son las librerías, estas se desarrollaron antes que los frameworks que se conocen actualmente. Su característica se basa en la utilización de métodos, clases, herencia y patrones de diseño más no en el control interno de los datos. Estos se enfocan en los flujos de datos que se procesa internamente, como ejemplo de librería podemos mencionar las siguientes: “Jquery y Mootools como librerías de Javacript y Twitter Bootstrap como librería de CSS” (Valdivia, 2016).

2.5. Entorno Frontend

Dentro de la creación de aplicaciones, a más de desarrollar proyectos con tecnologías avanzada, estos se basan en la interacción con el usuario, y también se usa mecanismos para diseñar como: Fireworks, Photoshop y otros. El objetivo de la tecnología Frontend es implementar aplicaciones que no requieren de una base de datos para poder ejecutarse, se enfocan en el diseño de interfaces y con el objetivo de proveer al usuario una experiencia adecuada para su funcionamiento.

2.6. Sistemas Operativos Móviles

Se define a un sistema operativo como aquel programa que es ejecutado en un dispositivo móvil. Este interactúa con el hardware para administrar y procesar los recursos según las necesidades. Los sistemas operativos para dispositivos móviles más populares en el mercado son: Android y IOS (Zatarain, 2019).

2.7. Android

Es un sistema operativo que se desarrolló como un pequeño proyecto que posteriormente fue adquirido por Google. En un inicio fue diseñado específicamente para celulares y en el trascurso del tiempo se ha implementado en televisores con conectividad a internet, GPS, artefactos de casa, etc.

“El núcleo de Android está basado en el de Linux para el manejo de memoria, procesos y hardware” (Zatarain, 2019).

2.8. IOS

Es un sistema operativo que ofrece Apple, está diseñado para telefonía celular. Es un software que permite la sincronización de archivos.

“iOS es un sistema operativo basado en Unix, el cual es una derivada del sistema operativo para computadores de Apple. A la vez MacOS está basado en el núcleo de Darwin BSD ” (Zatarain, 2019).

2.9. Aplicaciones Móviles

Una aplicación es una parte del software que funciona en un sistema operativo ya sea Android o iOS. Según Topón (2020) una aplicación móvil funciona dentro de un ecosistema en el cual se interconecta a dispositivos y otros productos de software.

2.10. Aplicaciones Nativas

Se llaman aplicaciones nativas porque se ejecutan a través de las funciones propias de los dispositivos móviles, esto depende del sistema operativo en el cual la aplicación se va a desarrollar. En este caso existirá la necesidad de utilizar un entorno y lenguaje de programación que sea compatible al sistema operativo del dispositivo.

Este tipo de aplicaciones se utilizan para mejorar el rendimiento de APIs, actualizaciones, representación de interfaz gráfica, además pueden resultar costosas porque son plataformas de especialización. “Son una mejor opción, si la facilidad de implementación y el tiempo de desarrollo es la principal preocupación del desarrollador. Pero se debe tener en cuenta que la aplicación no se puede publicar en el mercado de aplicaciones” (Shah et al., 2019).

2.11. Aplicaciones Hibridas

Este tipo de aplicaciones se diferencian de las nativas debido a que utilizan una capa de abstracción en un sistema operativo nativo. De esta manera permite el encapsulamiento de una aplicación en formato HTML. Esta tecnología hibrida permite reducir el tiempo de estimación y costes en una implementación de software móvil.

Además, puede conectarse a APIs nativas, pero suelen ser complejas para desarrollar. “Las aplicaciones híbridas combinan componentes nativos con el WebView primitivo que ofrecen las aplicaciones web” (Shah et al., 2019).

2.12. Aplicaciones Multiplataforma

Estas aplicaciones se diferencian de las anteriores porque logran ejecutar código desarrollado para la aplicación en múltiples fuentes. “Para la utilización de este tipo de aplicaciones hay que considerar varios factores como: la elección del SDK, la experiencia del usuario, la estabilidad del marco, la facilidad de actualización, el costo de desarrollo y el tiempo de comercialización de una aplicación” (Shah et al., 2019).

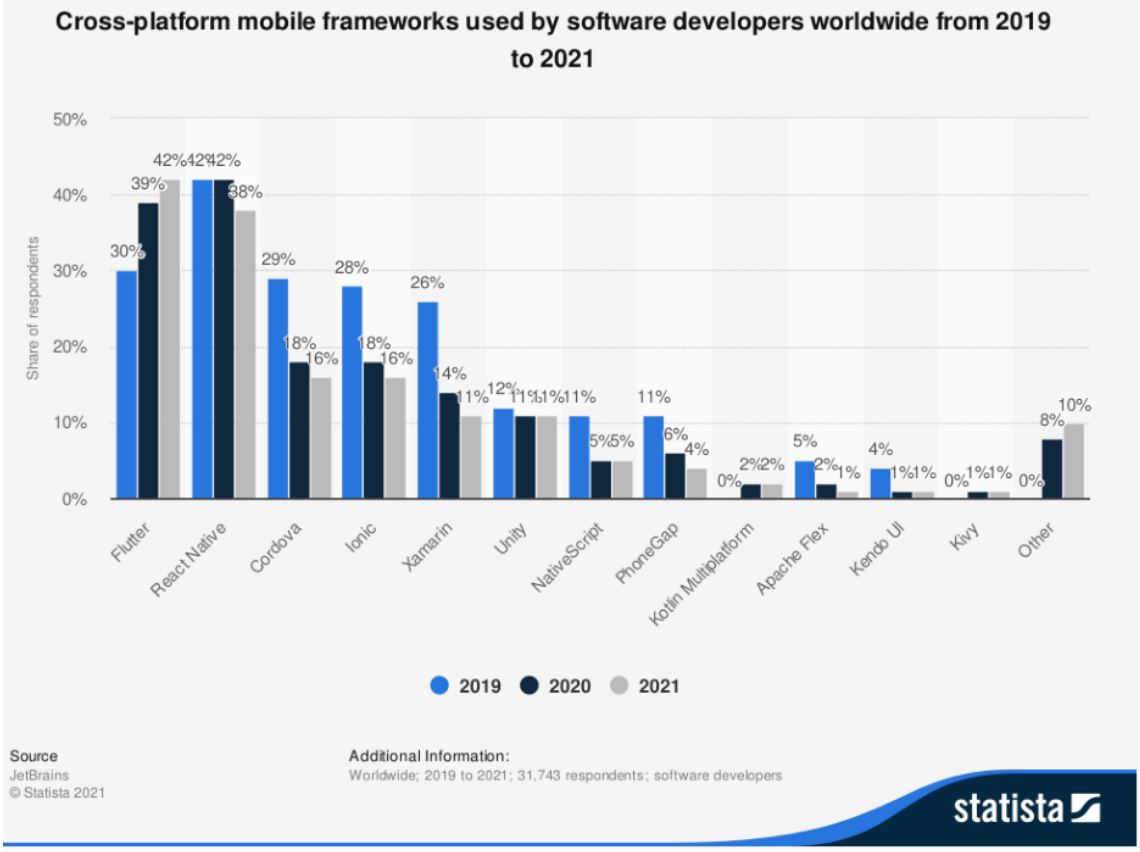
3. CAPITULO III: FRAMEWORKS MÓVILES

Para determinar los tipos de entornos que se van a analizar se utilizó datos estadísticos, que permiten conocer qué porcentaje de aceptación han tenido los diferentes frameworks que se encuentran en el mercado entre desarrolladores de software en todo el mundo.

Statista es una plataforma en línea que se dedica a la analítica de datos relacionados a varias temáticas de actualidad y posee información a través de opiniones y estudios de mercado en varios países. Recopila información de diferentes sitios web en los que se realizan encuestas o blogs como stack overflow, jetbrains, entre otros, y se analiza las respuestas de los usuarios (Terfehr, 2022).

A continuación, se presenta en la **Figura *2*** una gráfica de los entornos de desarrollo móvil que se está utilizando en el mercado.

Figura 2. *Estadísticas de entornos de desarrollo móvil en el mercado*



Fuente: (Terfehr, 2022)

Los datos presentados se obtuvieron a partir de la encuesta denominada “The State of Developer Ecosystem 2021”, realizada a una población de 31,743 personas en todo el mundo, se enfocó hacia desarrolladores de software entre los años 2019 al 2021.

De acuerdo con los datos analizados, se definió los principales frameworks móviles a investigar e identificar las características propias de cada uno. A continuación, se expone cada entorno con sus respectivas especificaciones.

3.1. Flutter

Es un entorno de desarrollo utilizado para interfaz gráfica, diseñado para aplicaciones móviles multiplataformas, fue creado por la empresa estadounidense Google. Utiliza el principio de código abierto (open source), el cual permite desarrollar aplicativos tanto para Android como para IOS. Este tipo de entorno se centra en la experiencia del usuario, el rendimiento de su interfaz y la reutilización de código utiliza el lenguaje de programación Dart, que también fue creado por Google, este lenguaje permite desarrollar la lógica en un aplicativo por lo que facilita el desarrollo de un proyecto.

La primera versión de este framework fue el 1.0, se lanzó en el año 2018 y cabe mencionar que a pesar de que era la primera versión esta ya tenía una cierta madurez, esto por la constante utilización de este tipo de entornos para el desarrollo de aplicativos que en la actualidad Google ofrece. También se conoce que la última actualización disponible en el mercado es la 2.2, esta proporciona un kit de herramientas para la construcción de novedosas aplicaciones, proporcionando al desarrollador una agradable y eficiente utilización de interfaces móviles.

Figura 3. *Logo de Flutter*

Fuente: (Flutter, 2021)

Para el desarrollo de la interfaz este entorno utiliza widgets que se han desarrollado mediante programación orientada a objetos, este funciona como un componente que puede ser personalizado dependiendo del requerimiento para el diseño. Una ventaja

importante es que el programador no se encuentra limitado por la estructura que maneja Flutter, es decir se puede crear componentes sin necesidad de programar el diseño desde cero, esto se da al momento de crear el ambiente, se genera código automáticamente como una plantilla que pueden ser modificado. Además, Flutter reacciona rápidamente a cambios ya que no es necesario recargar constantemente luego de modificar algún componente en la interfaz, se actualiza automáticamente el programa y muestra en tiempo real cada cambio.

Flutter tiene un funcionamiento de empaquetado similar a las aplicaciones nativas, También proporciona puntos como entradas para exista una comunicación con los servicios de la superficie de manera que se pueda presentar la interfaz adecuadamente. El código que se genera en este entorno puede ser adaptado a una aplicación que ya fue desarrollada.

3.1.1. Curva de Aprendizaje

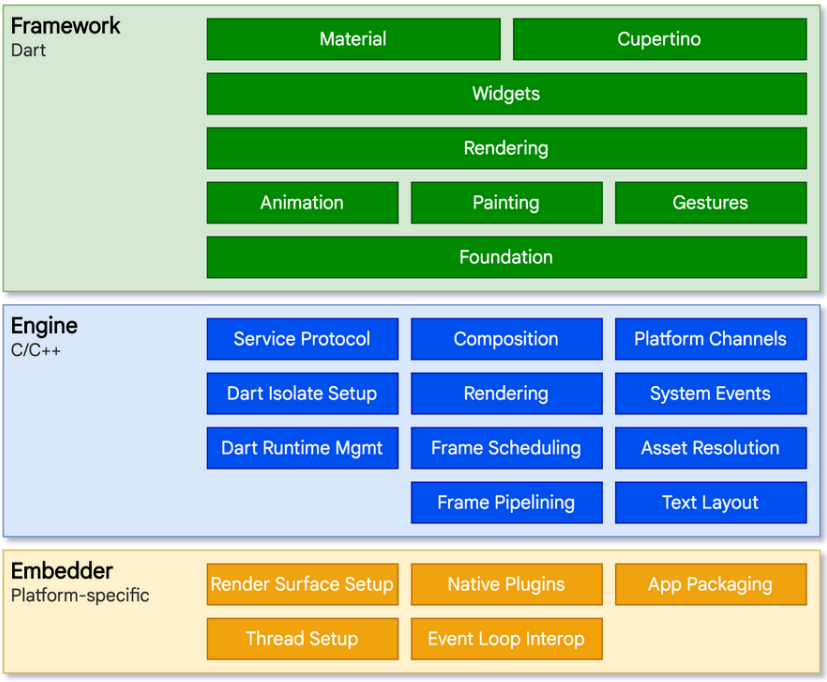
Otro de los aspectos que hacen de Flutter una herramienta sencilla de utilizar es la curva de aprendizaje, la cual al tener un ambiente con componentes disponibles y la utilización de un solo lenguaje de programación tanto para el backend como para el frontend disminuye el tiempo de implementación significativamente (Vázquez, 2019).

3.1.2. Arquitectura

Flutter está compuesto por capas en donde existe una variedad de bibliotecas que son independientes, cada una de estas es dependiente de la capa adyacente. Funciona de manera modular con la siguiente capa para poder realizar él envió de datos.

En la **Figura *4*** se muestra gráficamente la arquitectura de Flutter, se compone de 3 capas: la primera es la capa de la interfaz que utiliza el lenguaje de programación Dart, este permite desarrollar al programador los componentes y lógica del aplicativo; la segunda es la capa Engine está desarrollando en lenguaje C++; la tercera es la capa Embedder (incrustamiento) esta interactúa directamente con el hardware del dispositivo, permitiendo que las capas framework y engine se acoplen a los componentes e interactúen de manera muy similar a la de los lenguajes nativos (Flutter, 2021).

Figura 4. *Arquitectura de Flutter*



Fuente: *(Flutter, 2021)*

3.1.3. Integración

Este entorno tiene la ventaja de adaptarse a varias plataformas. A partir de la versión 1.20 se ha definido 3 niveles para dar soporte de aquellas plataformas para ejecutar Flutter. Existen 3 parámetros que son importantes para tener en cuenta, primero visualizar las plataformas que se han sido probadas por Google, luego se identifican aquellas que son probadas por la comunidad, finalmente las plataformas que no son soportadas corresponden a aquellas que en cuanto a funcionalidad se ejecutan correctamente pero que en el dispositivo de desarrollo no permite probar la funcionalidad. A continuación, en la **Tabla *2***, se muestra aquellas plataformas que tienen soporte para ejecutar este entorno.

Tabla 2. *Soporte de plataformas probadas por Google*

Versión Plataforma			
Android	Android SDK 30	IOS	14 (all)
Android	Android SDK 29	IOS	13.3-13-7
Android	Android SDK 28	IOS	13.0
Android	Android SDK 27	IOS	12.4 & 12.4.1
Android	Android SDK 26	IOS	9.3.6
Android	Android SDK 25		

Android	Android SDK 24
Android	Android SDK 23
Android	Android SDK 22
Android	Android SDK 21
Android	Android SDK 20
Android	Android SDK 19

Fuente: (Flutter, 2021)

3.1.4. Depuración

Existe herramientas que Flutter ofrece automáticamente al ejecutar la aplicación, Pero también hay la posibilidad de poder utilizar hot reload que significa recarga en caliente permitiendo así poder realizar cambios sin la necesidad de recargar toda la aplicación.

"Estas funcionalidades son proporcionadas por el mecanismo de flutter attach, este puede ser iniciado a través de diferentes vías, como a través de las herramientas CLI del SDK, a través de VS Code o IntelliJ/Android Studio" (Flutter, 2021).

3.1.5. Pruebas

Ofrece tres tipos de pruebas:

- Unidad de prueba: verifica únicamente las funciones, métodos o clases, el propósito es poder corregir la lógica bajo ciertas condiciones.
- Widget de prueba: también se le denomina prueba de componentes, esta verifica que la interfaz tenga una iteración correcta en cuanto a eventos, recepción de solicitudes y respuesta como se diseñó.
- Prueba de integración: verifica que todos componentes e interacciones funciones correctamente y de esta manera verificar el rendimiento del aplicativo.

A continuación, en la **Tabla *3*** se muestra la comparación entre estas pruebas.

Tabla 3. *Comparación entre tipos de pruebas*

	Unit	Widget	Integration
Confidence	Low	Higher	Highest
Maintenance cost	Low	Higher	Highest
Dependencies	Few	More	Most
Execution speed	Quick	Quick	Slow

Fuente: (Flutter, 2021)

3.1.6. Documentación

Proporciona una documentación medianamente variada, de manera que existen conceptos y guías de instalación, se menciona aspectos generales y posee poca información en el aspecto móvil. También por lo reciente del framework no existe aún una comunidad grande que aporte dudas para aquellos que están empezando a utilizar este entorno.

3.2. React Native

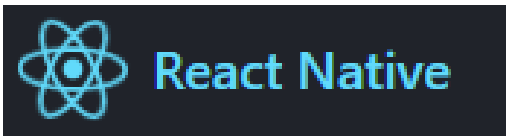
Es un marco de desarrollo para aplicaciones móviles multiplataforma, creado por Facebook, se lanzó como una versión específica de React en el 2015 y se ha mantenido hasta la actualidad. Inicialmente no figuraba como entorno open source hasta que se decidió dar apertura para que las comunidades enfocadas al desarrollo de interfaces pudieran hacerlo libremente. El objetivo principal de este framework es enfocarse en dar una experiencia de calidad al usuario y que a su vez ésta se ejecute en los sistemas operativos más utilizado en el mercado como son iOS y Android, sin la necesidad de requerir una visualización en la web de manera intermedia para la funcionalidad de la aplicación. Otro de los intereses que tiene este entorno es facilitar el trabajo a los desarrolladores, de manera que estos no necesiten adquirir un conocimiento total del framework, basta conocer la estructura básica para poder desarrollar una aplicación.

El lenguaje de programación que utiliza es JavaScript, “su núcleo se basa en React, en donde utiliza módulos nativos en vez del web”. Para la construcción de un producto de software ahora los desarrolladores web tienen la posibilidad de poder convertir sus

aplicaciones a móvil de manera que estas tengan un comportamiento nativo, esto se podría dar por medio de la biblioteca JavaScript.(Daisy & Imbaquingo, 2018).

Este entorno se encuentra respaldado principalmente por Facebook y ha sido implementado en varias aplicaciones populares de actualidad tales como, Instagram, Pinteres, Discord, Skype, Ubereats, entre otras.

Figura 5. *Logo React Native*



Fuente: (ReactNative, 2021b)

Para tener una idea más amplia del framework a continuación se especifica algunos parámetros importantes.

3.2.1. Rendimiento

Respecto a este punto hasta el momento no se garantiza totalmente un rendimiento que sea eficiente en todos los dispositivos móviles, ya que depende del tipo de aplicativo que se desea construir y el grado de complejidad que esta requiera, por lo que será necesario optimizar recursos de manera manual. El framework mejora ligeramente la interfaz de usuario, pero no garantiza un comportamiento óptimo general al desplegar la aplicación.

3.2.2. Actualización

Este entorno tiene la facilidad de permitir realizar cambios al mismo tiempo que la aplicación se ejecuta, este comportamiento es automático de manera que no es necesario recargar. El tiempo estimado que tarda en actualizar dependerá del recurso del dispositivo en el que se compile y se ejecute. Teniendo como estimado entre 3 y 4 segundos con un computador promedio.

3.2.3. Depuración

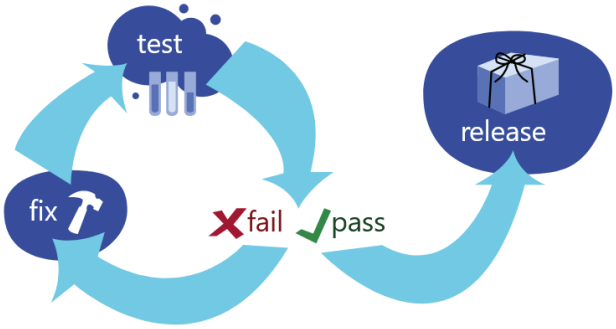
Para determinar de mejor manera el funcionamiento de la aplicación es importante probar en varios dispositivos reales antes de desplegar. React Native recomienda utilizar la herramienta Expo-Cli, la misma que proporciona un código QR para obtener una vista previa del aplicativo. También existe la posibilidad de depurar a través de un simulador que se adapte a Android como para iOS.

3.2.4. Pruebas

Los errores son una parte común que se puede encontrar al desarrollar y probar aplicaciones, estos errores se pueden dar por varias razones entre estas el mal uso por parte del usuario o fallos del sistema, en la **Figura *6***, se visualiza el proceso de pruebas.

Este entorno proporciona una manera automática para asegurar la calidad de los productos que se desarrollan, estas van desde un análisis estático hasta pruebas de extremo a extremo. Incluye herramientas de configuración de caja con análisis estático mismas que permite identificar eficacia del código, análisis de errores, utilización de código, comprobación de funcionalidad, de esta manera es posible disminuir el tiempo en esta fase.

Figura 6. *Proceso de pruebas de React Native*



Fuente: (ReactNative, 2021b)

3.2.5. Escalabilidad

Permite implementar nuevas funcionalidades a la aplicación que se desarrolló inicialmente, de manera rápida se pueden actualizar y acoplar nuevas dependencias al proyecto.

3.2.6. Versión

Las versiones son lanzadas mensualmente al iniciar un mes, a través de ramas al repositorio de GitHub en donde se crea posibles versiones que se mantienen disponibles para la comunidad de React Native en un mes, mismas que son probadas y verificadas por sus colaboradores los cuales tiene la posibilidad de reportar inconvenientes a través de informes, en este proceso existe la posibilidad de que la versión que se eligió se promueva a estable y de esta manera proyectarse a nuevos cambios para las siguientes

versiones que se lancen. La retro alimentación proporcionada por la comunidad es importante de manera que están constantemente conociendo la estabilidad del framework.

En la actualidad la última versión que se ha lanzado de este entorno es la 0.65 que se encuentra con su respectiva documentación.

3.2.7. Accesibilidad

En cuando a accesibilidad "Android como iOS proporcionan API para integrar aplicaciones con tecnologías de asistencia como los lectores de pantalla integrados VoiceOver (iOS) y TalkBack (Android). React Native cuenta con API complementarias que permiten que tu aplicación se adapte a todos los usuarios" (ReactNative, 2021a).

3.2.8. Documentación

React Native proporciona información bastante amplia para dar soporte a posibles cuestionamientos y dudas de los desarrolladores, incluso cuenta con visualizaciones animadas, ejemplos y preguntas frecuentes de la comunidad. Posee ayudas específicas de pendiendo de la versión que necesite documentación.

3.2.9. Seguridad

No contiene algún tipo de seguridad específico, pero existen sugerencias de herramientas o API's que pueden ser implementas, también proporciona una guía de buenas prácticas para seguridad que se menciona en la documentación.

3.3. Ionic

Es un framework de código abierto se utiliza para desarrollar aplicaciones híbridas, por tener esta tendencia de desarrollo utilizan tecnología web y se ejecuta en dispositivos que permite obtener un rendimiento alto de los dispositivos.

Para el desarrollo de la interfaz es importante indicar que renderiza los componentes al ejecutar la aplicación, se focaliza en la experiencia con el usuario para la integración de esta. También ofrece una biblioteca que permite la optimización de componentes, "los gestos, y herramientas para la construcción rápida, altamente aplicaciones interactivas" (Ionic, 2020).

El objetivo de este entorno es la implementación de aplicaciones en múltiples plataformas, ya sean estas para Android, iOS, de escritorio, nativas, web en una sola implementación de código.

Este entorno tiene servicios adicionales que son enfocado hacia el soporte empresarial, en donde se presentan soluciones con mejores funcionales, el inconveniente es que para acceder a estos requiere de un pago, lo que hace que este se limite. Dependerá mucho del ámbito en el que se desee utilizar este framework y los requerimientos que las empresas tengan.

Figura 7. *Logo Ionic*

Fuente: (Ionic, 2020)

3.3.1. Lenguaje de Programación

Este framework trabaja con el lenguaje JavaScript, lo que permite un amplio campo de implementación.

3.3.2. Rendimiento

En cuanto al rendimiento este funciona de manera adecuada en dispositivos actuales, en donde existe un aprovechamiento del hardware, las transiciones de la aplicación y la optimización del tacto. Para dar una ide del rendimiento de aplicaciones desarrolladas con este framework esta BMW, Nubank, Google assistant, entre otras.

3.3.3. Arquitectura

“Ionic CLI está construido con Manuscrito y Node.js . Es compatible con el Nodo 10.3+, pero el último Nodo LTS es siempre recomendable continúe con el desarrollo de open source en el repositorio de GitHub (Ionic, 2020).

3.3.4. Compatibilidad de entornos

Está diseñado para poder integrarse con la mayoría de las plataformas correctamente, entre estos se encuentra Vue, Angular, React, Java Script, lo que ha permitido en el marco sea más flexible y adaptativo a muchas plataformas en donde se desarrolla aplicaciones.

3.3.5. Soporte

Este entorno se encuentra dando soporte y mantenimiento todo el tiempo por un equipo de ionic, Tiene una amplia comunidad internacional lo que permite tener un feedback de colaboradores que se dedican a al crecimiento del entorno. También cuenta con soporte en el ámbito empresarial, de manera que los grandes y pequeños desarrolladores comparten aplicaciones que se ejecutan en cualquier plataforma.

3.3.6. Licencia

Por ofrecer un entorno de código abierto, se ha liberado licencias MIT es decir se desconocer los derechos del autor y se concede permisos sin restricción para la construcción de proyectos. Trabaja con la licencia 2.0 de Apache, provee confiabilidad y larga duración de los productos de software.

3.3.7. Curva de aprendizaje

Una de las características de Ionic es la simplicidad del entorno, de manera que la implementación de aplicaciones es fácil de aprender y accesible para cualquier tipo de persona que tenga conocimiento en desarrollo web.

3.3.8. Documentación

Cuenta con guías de instalación, utilización de herramientas, conceptos enfocados a cada tipo de plataforma por lo que provee un soporte completo, además cuenta con una comunidad grande que permite mantenerse actualizado en experiencias de usuarios en varios pases. A demás cuenta con traducción de documentación, hasta el momento se cuenta con traducciones de inglés japones, chino, frances, portugués y español.

3.4. Xamarin

Es un entorno de código abierto que permite ejecutar aplicaciones actuales con un buen rendimiento para iOS, Android, .Net de Windows. Se ejecuta a través de un entorno de administración el mismo que permite obtener ventajas en cuanto a la memoria y recopilación de información que no se ha utilizado.

3.4.1. Lenguaje de Programación

Este framework utiliza el lenguaje de programación C# como base y se puede utilizar el IDE oficial Visual Studio Code proporcionado por Microsoft como herramienta de desarrollo para las aplicaciones.

3.4.2. Soporte

A inicios de 2017 Xamarin fue la plataforma de desarrollo móvil más utilizada debido a la posibilidad que ofrecía para crear aplicaciones híbridas, algo relativamente nuevo en ese entonces. Con el paso de los años, la comunidad de desarrolladores se ha orientado por otros frameworks, siendo Flutter el más popular actualmente.

3.4.3. Licencia

Xamarin es una plataforma Open Source y de código abierto en la cual se puede crear aplicaciones para Android e iOS, además nos ofrece la posibilidad de compilarlas para el SO Windows.

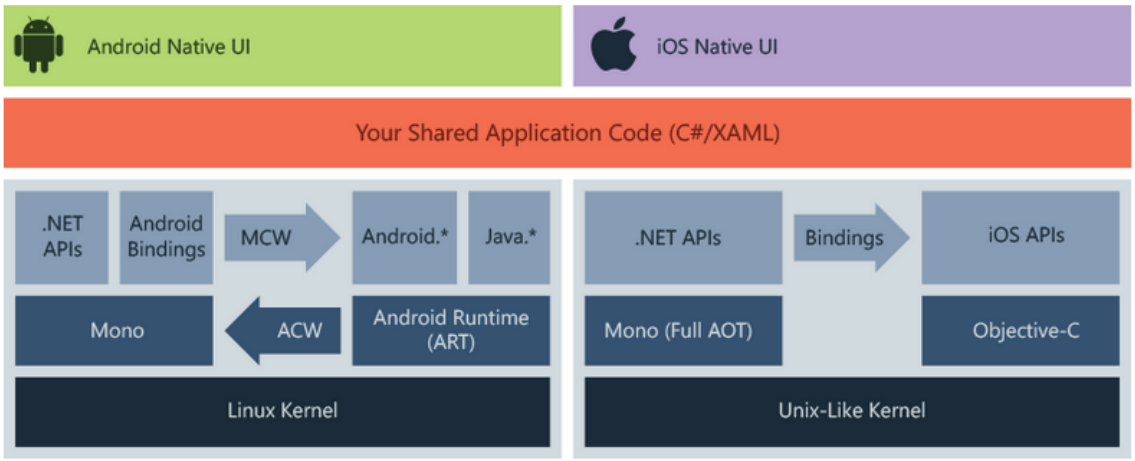
3.4.4. Sistema operativo

Xamarin es una plataforma hibrida que ofrece la posibilidad de crear aplicaciones compatibles en un 80% con Android e iOS, permitiendo escribir la lógica del negocio en un solo lenguaje y configurar la presentación de los componentes visuales por separado para cada plataforma. Además, cuenta con la capacidad de compilar aplicaciones para el sistema operativo Windows haciendo uso de .NET.

3.4.5. Arquitectura

Xamarin utiliza una arquitectura en capas que se comunican con las adyacentes dentro de un entorno administrado, en la **Figura *8***, se muestra la arquitectura. Esto permite mejorar aspectos clave como la asignación de memoria y el reciclaje de elementos no utilizados.

Figura 8. *Arquitectura de Xamarin*



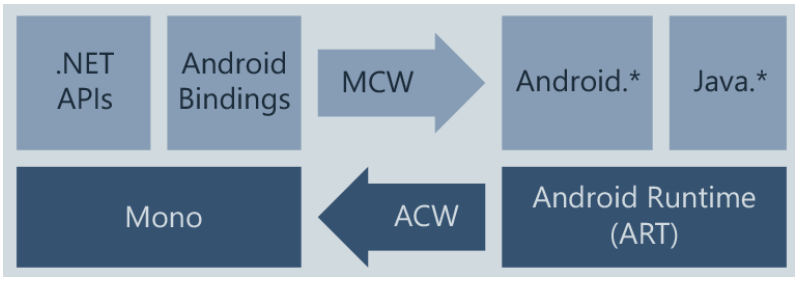
Fuente: (Xamarin, 2021)

Como se puede ver en el gráfico anterior con este framework se puede mantener un mismo código para la lógica del negocio y a partir de ello crear módulos de visualización con características nativas de cada lenguaje como se mostrará a continuación.

3.4.6. Android

Las aplicaciones dirigidas hacia Android utilizan como lenguaje intermedio C#, que en la siguiente etapa pasa a un ensamblado nativo cuando la aplicación inicia. Estas aplicaciones se ejecutan en el entorno de ejecución MONO. En la **Figura *9***, se muestra la arquitectura

Figura 9. *Arquitectura Android*

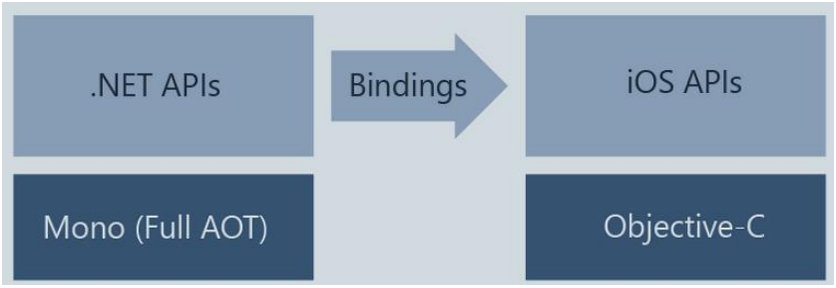


Fuente: (Xamarin, 2021)

3.4.7. iOS

Por otro lado, las aplicaciones orientadas a iOS se compilan de manera anticipada (AOT) pasando desde el lenguaje C# hacia ensamblado nativo ARM a través de selectores que permiten exponer el lenguaje de programación Objective-C, en la **Figura 10** se muestra la arquitectura.

Figura 10. *Arquitectura iOS*



Fuente: (Xamarin, 2021)

3.4.8. Rendimiento

A más de aportar al desarrollo de aplicaciones nativas, tiene la capacidad para acceder a las funcionalidades de los dispositivos de manera que el hardware se acelere y se aproveche en su totalidad el rendimiento nativo.

3.4.9. Escalabilidad

Debido a que la implementación de nuevas funcionalidades es muy frecuente en aplicaciones, Xamarin Azure proporciona soluciones que sean escalables y eficientes. Ofrece servicios de almacenamiento, base de datos, servicios inteligentes, servidor y la nube, estos pueden ser complementos que añaden a las funcionalidades sin tener la necesidad de requerir a otras empresas para complementar a los aplicativos.

3.4.10. Documentación

Cuenta con información para cada sistema operativo al cual se desea implementar, también tiene guías de instalación y es interactiva de manera que contiene videos para complementar la información que ofrece el entorno.

3.5. Cordova

Es un marco para desarrollo de aplicaciones móviles, permitiendo la utilización de tecnologías web y el lenguaje de JavaScrip para múltiples plataformas, de esta manera es posible evitar desarrollar código nativo para cada plataforma. “Las aplicaciones se ejecutan dentro de envolturas para cada plataforma y dependen de enlaces estándares API para acceder a de cada dispositivo sensores, datos y estado de la red.”

Figura 11. *Logo Cordova*

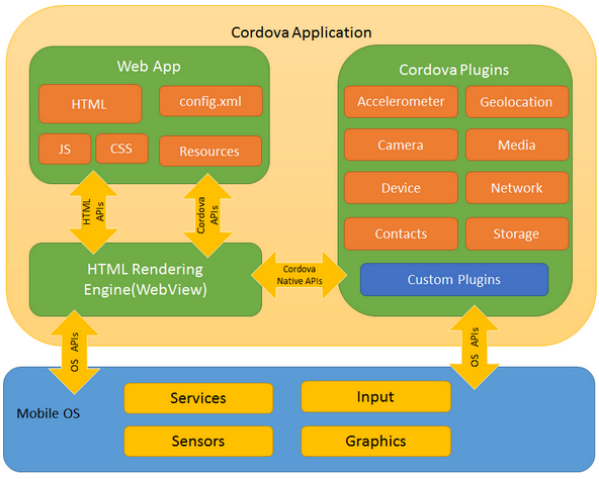
Fuente: (Cordova, 2021)

3.5.1. Arquitectura

Se basa en archivos que proporcionan información de la aplicación como es config.xml. Cada aplicación que se crea se ejecuta a través de webView dentro de la capa nativa que posterior será distribuida a las tiendas de cada sistema operativo.

WebView Cordova permite a la aplicación construir una interfaz completa para el usuario, en algunos casos esta también funciona como un componente dentro de una app hibrida que es más grande. En la **Figura *12*** se visualiza la arquitectura, al utilizar este entorno se tiene ventajas como la reutilización de código, disminución de herramientas y el lenguaje que corresponda a la plataforma en el cual se va a desarrollar.

Figura 12. *Arquitectura de Cordova*



Fuente: (Cordova, 2021)

3.5.2. Versionamiento

Tiene una variedad de versiones que se han desarrollada para una mejorar del entorno, a partir del lanzamiento de la versión 4.3.0 en adelante córdoba permite la restauración de plataformas, plugins y almacenar la aplicación. Hasta el momento se encuentra con la versión más reciente es 10X.

3.5.3. Seguridad

Es un aspecto necesario de manera que al interactuar con datos existe la posibilidad de ser vulnerados, Cordova menciona algunas características en donde este entorno sugiere algunas recomendaciones importantes.

Por naturaleza a las aplicaciones que navegan libremente se recomienda que se restrinja el acceso solo para dominios confiables. También si el contenido se aprovecha a través de un iframe por medio de un dominio este puede acceder a las funciones nativas de córdoba, pero existe la posibilidad de ser vulnerado por lo cual no es tan recomendable la utilización de esos.

“Cordova no admite certificado verdadera fijación. La barrera principal para esto es una falta de API nativas en Android para interceptar conexiones SSL para realizar la comprobación del certificado del servidor” (Cordova, 2021).

3.5.4. Soporte

Cordova presenta una comparación de herramientas de desarrollo y para dispositivos móviles, en la

Figura 13 se muestran las plataformas versus sistemas operativos, se encuentran disponibles con su documentación específica.

Figura 13. Soporte de plataformas en Cordova

	Amazon-fireos	Android	blackberry10	Firefox OS	Ios	Ubuntu	WP8 (Windows Phone 8)	Windows (8.0, 8.1, 10, Teléfono 8.1)	Tizen
cordova CLI	✓ Mac, Windows, Linux	✓ Mac, Windows, Linux	✓ Mac, Windows	✓ Mac, Windows, Linux	✓ Mac	✓ Ubuntu	✓ Windows	✓	✗
Embedded WebView	✓ (ver detalles)	✓ (ver detalles)	✗	✗	✓ (ver detalles)	✓	✗	✗	✗
Plugin Interface	✓ (ver detalles)	✓ (ver detalles)	✓ (ver detalles)	✗	✓ (ver detalles)	✓	✓ (ver detalles)	✓	✗
	API de la plataforma								
Acelerómetro	✓	✓	✓	✓	✓	✓	✓	✓	✓
HomeScreen PlusPlus	✓	✓	✓	✓	✓	✗	✓	✓ * Windows Phone 8.1 sólo	✓
Cámara	✓	✓	✓	✓	✓	✓	✓	✓	✓
Captura	✓	✓	✓	✗	✓	✓	✓	✓	✗
Brújula	✓	✓	✓	✗	✓ (3GS+)	✓	✓	✓	✓
Conexión	✓	✓	✓	✗	✓	✓	✓	✓	✓
Contactos	✓	✓	✓	✓	✓	✓	✓	parcialmente	✗
Dispositivo	✓	✓	✓	✓	✓	✓	✓	✓	✓
Eventos	✓	✓	✓	✗	✓	✓	✓	✓	✓
Archivo	✓	✓	✓	✗	✓	✓	✓	✓	✗

Transferencia de archivos	✓	✓	✓ * No apoyan onprogress ni abortar	✗	✓	✗	✓ * No apoyan onprogress ni abortar	✓ * No apoyan onprogress ni abortar	✗
Geolocalización	✓	✓	✓	✓	✓	✓	✓	✓	✓
Globalización	✓	✓	✓	✗	✓	✓	✓	✓	✗
InAppBrowser	✓	✓	✓	✗	✓	✓	✓	utiliza el iframe	✗
Los medios de comunicación	✓	✓	✓	✗	✓	✓	✓	✓	✓
Notificación	✓	✓	✓	✗	✓	✓	✓	✓	✓
SplashScreen	✓	✓	✓	✗	✓	✓	✓	✓	✗
Barra de estado	✗	✓	✗	✗	✓	✗	✓	✓ Windows Phone 8.1 sólo	✗
Almacenamiento de información	✓	✓	✓	✗	✓	✓	✓ localStorage & indexedDB	✓ localStorage & indexedDB	✓
Vibración	✓	✓	✓	✓	✓	✗	✓	✓ * Windows Phone 8.1 sólo	✗

Fuente: (Cordova, 2021)

3.6. Documentación

Ofrece guías de instalación, recomendaciones para la utilización de herramientas, blogs permitiendo la colaboración de nuevo conocimiento. Contiene información de las plataformas en las cueles se puede desarrollar aplicaciones con sus respetivas características.

4. CAPÍTULO IV: ANÁLISIS COMPARATIVO DE LOS FRAMEWORKS DE DESARROLLO PARA APLICACIONES MÓVILES.

4.1. Análisis comparativo entre los diferentes frameworks

A continuación, se presenta tablas comparativas entre los frameworks escogidos de acuerdo con las características propias de cada uno. Para poder obtener un análisis claro de que entorno tiene más ventajas que otro se utilizará la siguiente abreviatura.

Alto: El framework cumple con el parámetro especificado.

Medio: Parámetro que depende de otros factores para poder definirse.

Bajo: No cumple con el parámetro adecuadamente.

ALTO = A

MEDIO = M

BAJO = B

En las tablas que se presentan a continuación se puede identificar las ventajas y desventajas que tienen unos con otros.

4.2. Flutter vs Xamarin

En la **Tabla *4***, se detallan cada una de las características de los entornos Xamarin y Flutter.

Tabla 4. *Comparación de características entre Flutter y Xamarin*

CARACTERÍSTICAS	XAMARIN	FLUTTER
Arquitectura	A	A
Rendimiento	A	A
Lenguaje de programación	A	A
Curva de aprendizaje	M	M
Precio/Open Source	A	A
Depuración	A	A
Pruebas	B	A
Soporte de plataformas	B	A
Escalabilidad	A	A
Versionamiento	A	A
Seguridad	B	B
Madurez	A	M
Reutilización	A	M
Documentación	A	A

Análisis:

La arquitectura de los dos entornos tiene la capacidad para poder desarrollar un aplicativo móvil, en cuanto a rendimiento ambos tienen la capacidad para hacer de un proyecto productivo, el lenguaje de programación es estable, la curva de aprendizaje tiene su variabilidad debido a que cuenta con varias librerías y APIs para el desarrollo, pero en el momento de implementar algo muy específico este requiere de una herramienta adicional lo que ocasiona que se prolongue en tiempo de aprendizaje.

Futter tienen un inconveniente y es que al utilizar APIs nativas en un código nativo este incrementa el tiempo de aprendizaje, debido a que sería necesario aprender otros lenguajes de programación para poder adaptar al entorno, la depuración en los dos casos utiliza un hot-reload que se refiere a recarga en caliente.

El soporte de plataformas de Xamarin permite desarrollar aplicaciones para Android, iOS, plataformas universales de Windows, pero para el sistema operativo de iOS es necesario de un equipo de Mac para poder implementar una aplicación, flutter por otra parte soporta más plataformas.

El versionamiento de cada entorno se va actualizando de acuerdo con el avance de madurez que va adquiriendo el framework, para poder determinar la escalabilidad se ha analizado que los dos entornos están abiertos a la reutilización de código y a la integración de nuevas funcionalidades. Respecto a la seguridad no se encuentra aún información que sustente el mecanismo de funcionamiento de estos entornos.

Cada framework tiene sus ventajas y desventajas, pero Xamarin por su antigüedad tiene una leve ventaja contra Flutter en cuanto a madurez, debido a que Xamarin utiliza un lenguaje de programación estable, en comparación con flutter. Flutter a pesar de ser un ambiente reciente, este se basa netamente en un único lenguaje de programación por lo que no todas las funcionalidades podrían adaptarse adecuadamente a otros entornos.

Finalmente, los dos se encuentran apoyados por grandes comunidades que aportan con experiencias de los usuarios que trabajan con estos frameworks y la documentación que ofrecen permite conocer con más detalle cada funcionalidad para poder implementar el ambiente que se requiere.

4.3. Ionic vs Xamarin

En la **Tabla *5***, se detallan cada una de las características de los entornos Ionic y Xamarin.

Tabla 5. *Comparación de características entre Ionic vs Xamarin*

CARACTERÍSTICAS	IONIC	XAMARIN
Arquitectura	A	A
Rendimiento	A	A
Lenguaje de programación	A	A
Curva de aprendizaje	A	M
Precio/Open source	M	A
Depuración	A	A
Pruebas	A	B
Soporte de plataformas	A	A
Escalabilidad	A	A
Versionamiento	A	A
Seguridad	A	B
Madurez	A	A
Reutilización	A	A
Documentación	A	A

Análisis:

En cuanto a la arquitectura los dos permiten desarrollar aplicaciones hibridas, generando un buen rendimiento en los dispositivos que se ejecute. Ionic utiliza el lenguaje de programación JavaScript mientras que Xamarin funciona con C#, estos dos lenguajes se mantienen estables para los entornos.

Cada framework utiliza diferentes lenguajes, pero se puede decir que Ionic al trabajar con JavaScript tiene una gran ventaja, y es que tiene un campo amplio para migrar hacia otros entornos que también trabajan con este lenguaje, a diferencia de Xamarin que trabaja con c#, el mismo que se ocupa solo para este entorno. A demás se puede decir que el tiempo de aprendizaje sería mejor aprovechado, ya que se podría aprender una sola vez el lenguaje de programación y tendría la facilidad de utilizar otros frameworks similares como React lo que no sucedería en el caso de Xamarín.

Al momento de implementar otras funcionalidades, los dos ambientes están en la capacidad de hacer de una aplicación escalable y adaptable a las necesidades que se requieran, los dos manejan código abierto, pero Ionic ofrece a las empresas más

funcionalidades como almacenamiento en la nube, seguridad, plugin de acceso entre otras características para soportar las necesidades empresariales.

Los dos cuentan con comunidades que aportan conocimientos y una documentación interactiva que va desde la instalación hasta la utilización de componentes. Además, Ionic tiene la ventaja de proporcionar adicionalmente soporte personalizado por el equipo de este entorno.

4.4. Ionic vs Flutter

En la **Tabla *6***, se detallan cada una de las características de los entornos Ionic y Flutter.

Tabla 6. *Comparación de características entre Ionic y Flutter*

CARACTERÍSTICAS	IONIC	FLUTTER
Arquitectura	A	A
Rendimiento	M	A
Lenguaje de programación	A	A
Curva de aprendizaje	A	M
Precio/Open source	M	A
Depuración	A	A
Pruebas	A	A
Soporte de plataformas	A	A
Escalabilidad	A	A
Versionamiento	M	M
Seguridad	B	B
Madurez	A	M
Reutilización	A	M
Documentación	A	A

Análisis:

Las dos arquitecturas soportan el desarrollo de aplicaciones móviles. En cuanto al rendimiento Ionic funciona de manera adecuada en los dispositivos actuales, en aquellos que sean antiguos existe la posibilidad de tener inconvenientes y por lo tanto no se ejecuten las aplicaciones adecuadamente. Al contrario de Flutter que este se adapta a las funcionalidades nativas del dispositivo haciendo que las animaciones o transiciones puedan ejecutarse de manera óptima.

La curva de aprendizaje para Ionic es relativamente baja, debido a que se ofrece un entorno sencillo y reduce la dificultad en el proceso de aprendizaje. Además, es importante tener en cuenta que cualquier persona que tenga conocimiento para desarrollar

una página web se encuentra en capacidad de poder implementar una aplicación con este entorno.

Flutter por otro lado se ha dicho que al utilizar un solo lenguaje de programación facilita el aprendizaje, pero aún no es tan flexible, es decir al momento de necesitar adaptarse a otras plataformas este requeriría de herramientas adicionales lo que incrementaría el tiempo de aprendizaje. En cuanto al costo del framework, flutter e Ionic ofrecen entornos de código abierto, con la diferencia de que Ionic ofrece adicionalmente tres tipos de planes que se consolidan al ámbito empresarial.

Los dos utilizan la ejecución en caliente para depurar la interfaz de usuario, lo que permite que cualquier cambio realizado se actualice automáticamente. Con respecto a la escalabilidad, flutter se adapta a cambios debido a que el lenguaje de programación puede ser utilizado en el backend de la aplicación, Ionic también permite la adaptación de cambios en cuanto la aplicación lo requiera.

Ambos frameworks están apoyados por sus comunidades, lo que permite que exista una retroalimentación de nuevas maneras de utilizar ciertos componentes, y estructuras que se adaptan de la experiencia de otros usuarios.

4.5. React vs Ionic

En la **Tabla *7***, se detallan cada una de las características de los entornos React e Ionic

Tabla 7. *Comparación de características entre React e Ionic*

CARACTERÍSTICAS	REACT	IONIC
Arquitectura	A	A
Rendimiento	M	A
Lenguaje de programación	A	A
Curva de aprendizaje	M	A
Precio/Open source	A	M
Depuración	A	A
Pruebas	A	A
Soporte de plataformas	A	A
Escalabilidad	A	A
Versionamiento	A	A
Seguridad	B	A
Madurez	A	A
Reutilización	A	A
Documentación	A	M

Análisis:

Estos entornos tienen varias características en común entre estas el lenguaje de programación como es Javascript, lo que permite que se adapten a otros entornos de desarrollo sin inconvenientes. La arquitectura cuenta con los componentes adecuados para poder desarrollar una aplicación móvil.

El rendimiento de Ionic permite utilizar eficientemente las funcionalidades nativas de los dispositivos, cosa que no ocurre con React, debido a que este no garantiza totalmente la eficiencia de los dispositivos, ya que dependerá del tipo de aplicación que se vaya a desarrollar y las especificaciones que requiera. La depuración que manejan los dos frameworks es la ejecución en caliente lo que permite visualizar automáticamente los cambios.

En cuanto al aprendizaje para estos dos ambientes será necesario comprender un solo lenguaje de programación lo que permitirá que se reduzca el tiempo de aprendizaje en el manejo del framework, como la adaptación a otras plataformas. El código generado se puede reutilizar en un gran porcentaje, lo cual es una gran ventaja de este punto, además al momento de implementar nuevas funcionalidades estas podrán adaptarse rápidamente por medio de dependencias agregadas al proyecto.

Estos entornos cuentan con comunidades que apoyan con conocimientos válidos. En el caso de Rect la comunidad es partícipe de los lanzamientos de nuevas versiones, siendo esto de gran ayuda para una mejora continua del entorno. La documentación es detallada para cada sistema operativo, en Ionic se presenta una documentación un poco limitada, pero con la ventaja de que proporcionan soporte personalizado para aquellos que adquieren un plan adicional.

4.6. Ionic vs Cordova

En la **Tabla *8***, se detallan cada una de las características de los entornos Ionic y Cordova

Tabla 8. *Comparación de características entre Ionic vs Cordova*

CARACTERÍSTICAS	IONIC	CORDOVA
Arquitectura	A	A
Rendimiento	A	A
Lenguaje de programación	A	A
Curva de aprendizaje	A	A
Precio/Open source	M	A
Depuración	A	A

Pruebas	A	A
Soporte de plataformas	A	A
Escalabilidad	A	A
Versionamiento	A	A
Seguridad	A	B
Madurez	A	A
Reutilización	A	A
Documentación	M	M

Análisis:

Estos dos entornos al igual que los anteriores utilizan el lenguaje de programación JavaScript lo que permite que exista compatibilidad con otros entornos. Tanto Ionic como Cordova son frameworks multiplataforma que permiten generar aplicaciones híbridas.

Ambos cuentan con arquitecturas que soportan los sistemas operativos de iOS y Android, permiten la construcción completa de interfaz para el usuario. El código que se genera puede ser reutilizado y adaptado a otros entornos que trabajan con JavaScript. Los dos manejan código abierto. Ambos son compatibles con otras instancias, pero en el caso de Cordova tiene un campo más amplio de plataformas como: Amazon-fireos, BlackBerry, Ubuntu, Windows, haciendo de este entorno uno de los más adaptables.

En cuanto a la depuración Ionic y Cordova utilizan la recarga en caliente para hacer que el entorno sea productivo. Además, estos entornos han ido madurando a través de los últimos años.

Poseen documentación didáctica, en el caso de Ionic está apoyado por una comunidad que prueba el entorno y da soporte personalizado. En Cordova a más de presentar detalle de como instalar el entorno y como utilizar componentes, contienen referencias y especificaciones para cada plataforma lo que hace que se tenga información para poder entender su funcionamiento.

Cordova no ofrece una herramienta para seguridad, pero hace referencias teóricas y recomendaciones sobre que se podría hacer en casos específicos con el framework. Ionic en cambio sí ofrece seguridad a través de otras herramientas, pero estas tienen un costo adicional.

4.7. Comparación entre los entornos

En la **Tabla *9*** a continuación, se realiza una comparación de manera general cada una de las características lo que permite analizar cómo se encuentran valorados los entornos y que beneficios aportan unos contra otros.

Tabla 9. *Comparativa de las características entre todos los entornos*

CARACTERÍSTICAS	IONIC	REACT	FLUTTER	XAMARIN	CORDOVA
Arquitectura	A	A	A	A	A
Rendimiento	A	M	A	M	A
Lenguaje de programación	A	A	A	A	A
Curva de aprendizaje	A	M	A	A	A
Precio/Open source	M	A	A	M	A
Depuración	A	A	A	A	A
Pruebas	A	A	A	A	A
Soporte de plataformas	A	A	M	A	A
Escalabilidad	A	A	A	A	A
Versionamiento	A	A	A	A	A
Seguridad	A	B	B	A	B
Madurez	A	A	M	A	A
Reutilización	A	A	A	A	A
Documentación	M	A	M	M	M

Análisis:

Cada entorno tiene ventajas y desventajas que dependen de la planificación y requerimientos de los desarrolladores ya sea que estos trabajen de manera individual, pertenezcan a un equipo de trabajo o a una empresa.

Mediante este análisis se ha logrado identificar varios aspectos que influyen en esta decisión y es que no existe como tal una definición concreta del campo en el cual se puede utilizar uno u otro, todos los entornos pueden ser implementados en varios campos de aplicación. Pero es necesario exponer que Ionic y Xamarín a más de brindar un ambiente, estos ofrecen varios servicios adicionales que se enfocan ya en la parte de la arquitectura y como tal en aspectos empresariales.

Es debido a esto que para poder determinar de una manera más clara como escoger un entorno, es a través de los parámetros que se han planteado. Es decir, que dependiendo de que parámetros son los más importantes se puede ir descartando uno de otro y determinar cuál se adapta mejor a las necesidades y visión del proyecto a implementar.

5. CAPÍTULO V: DISEÑO Y DESARROLLO

5.1. Tamaño de la muestra

Para definir el número de la muestra que se necesita para la recolección de información, se utilizó la siguiente fórmula:

$$n = \frac{N\sigma^2 Z^2}{(N-1)e^2 + \sigma^2 Z^2},$$

En la **Tabla *10***, se describen cada uno de los términos que compone la fórmula. Para el tamaño de la población se tomó el dato referencial de usuarios del sitio web stack overflow que son 50 millones.

Tabla 10. *Descripción de cada término*

Variable	Descripción	Datos
N	Tamaño de la población	50000000
Z	Valor de la distribución normal para un nivel de confianza del 95%	1,96
σ	Desviación estándar de la población	0,5
e	Error aceptable	0,05

$$n = \frac{N * 0{,}5^2(1{,}96)^2}{(N-1)(0{,}05)^2 + (0{,}5)^2 1{,}96^2}$$

$$n = 385{,}120477$$

$$n = 385$$

Luego de determinar el número de muestras necesarias se realizó encuestas, orientado a una población que se encuentra desarrollando o ya ha desarrollado aplicaciones móviles, esto con la finalidad de poder conocer las preferencias que tienen referente a los cinco tipos de entornos que se está analizando en esta investigación y de esta manera minimizar el error en los datos.

5.2. Recolección de Data para implementar el modelo de Aprendizaje Supervisado

Para el proceso de recolección de datos fue importante tomar en cuenta el tamaño de población a la cual se desea realizar el análisis, y a partir de este dato se determinó con cuantas encuestas se requiere para poder realizar un análisis comparativo con un margen de error del 5%. En el **Anexo *1***, se detalla la encuesta que se utilizó.

Los datos de la población se tomaron del sitio web Stack Overflow, el mismo que se dedica a la obtención de datos estadísticos de una gran cantidad usuarios de 126 países, este sitio está enfocado en la programación y desarrollo de proyectos en diversos lenguajes de programación y entornos, por lo que se ha considerado que es un lugar confiable para la obtención de datos.

5.3. Depuración de la data recolectada en las encuestas

Para el desarrollo del cuestionario se utilizó una escala de clasificación la cual permite obtener una retroalimentación de la población encuestada en forma comparativa, cada encuestado ha podido calificar varias características de los entornos de manera imperceptible. Al establecer este tipo de encuesta, los datos que se obtuvieron fueron transformados y depurados.

Durante el proceso de transformación de datos, se utilizó una escala gráfica de clasificación, la cual permite colocar un rango numérico a cada variable de respuesta. Los valores asignados se encuentran representados en la **¡Error! No se encuentra el origen de la referencia.**.

5.4. Análisis del modelo de aprendizaje supervisado para la realización del sitio web

Existen dos maneras de entrenar a una máquina de aprendizaje, puede ser mediante: El aprendizaje supervisado que utiliza datos etiquetados, este sirve para hacer modelos de minería de datos predictiva, es decir se entrena un modelo y a partir de este entrenamiento se puede predecir ciertos valores a futuro. Por otro lado, se encuentra el aprendizaje no supervisado que utiliza datos no etiquetados y sirve para realizar minería de datos descriptiva, de manera que permite agrupar y describir una situación que se desea plantear, este tipo de análisis ya dependerá del analista de datos.

Para el desarrollo de este trabajo se utilizará el aprendizaje supervisado el cual contiene dos grandes grupos de algoritmos de aprendizaje como regresiones y clasificación.

- **Regresiones (Univariable y Multivariable)**
  - Regresión Lineal
  - Ridge
  - Regresión de Vecinos Cercanos (KNN)
  - Redes Neuronales
- **Clasificación**
  - Vecinos Cercanos (KNN)
  - Árboles de decisión
  - Redes Neuronales

Para este proyecto se analizará los modelos de clasificación, ya que se ha tomado en cuenta el tipo de dato cualitativo que permite categorizar, y representar las características de los entornos de desarrollo que se están evaluando. En cuanto a las regresiones, estas no cumplen con el tipo de análisis que se desea desarrollar, debido a que utiliza un tipo de dato continuo.

A partir de los tipos de algoritmos de clasificación se pretende analizar cuál de estos presenta los mejores resultados, considerando el rendimiento, el porcentaje de entrenamiento y la correlación de los datos.

5.5. Vecinos Cercanos (KNN)

Es un clasificador que se basa tomando en cuenta la vecindad que existe en el proceso de aprendizaje, esto permite que el algoritmo sea simple y eficiente dentro de la clasificación de data. En la **Figura *14***, se muestra las etapas de modelamiento que utiliza KNN. En este clasificador se evalúa el parámetro K el cual hace referencia a la cantidad de vecinos con los que se define a que categoría le pertenece, este valor se define de manera práctica, es decir al probar el algoritmo se testea qué valor funciona mejor. También es importante definir la distancia que existe entre cada vecindad con cada elemento analizar del conjunto de datos (De La Hoz et al., 2019).

Figura 14. *Modelo de clasificación KNN para predicción de entornos de desarrollo móvil*

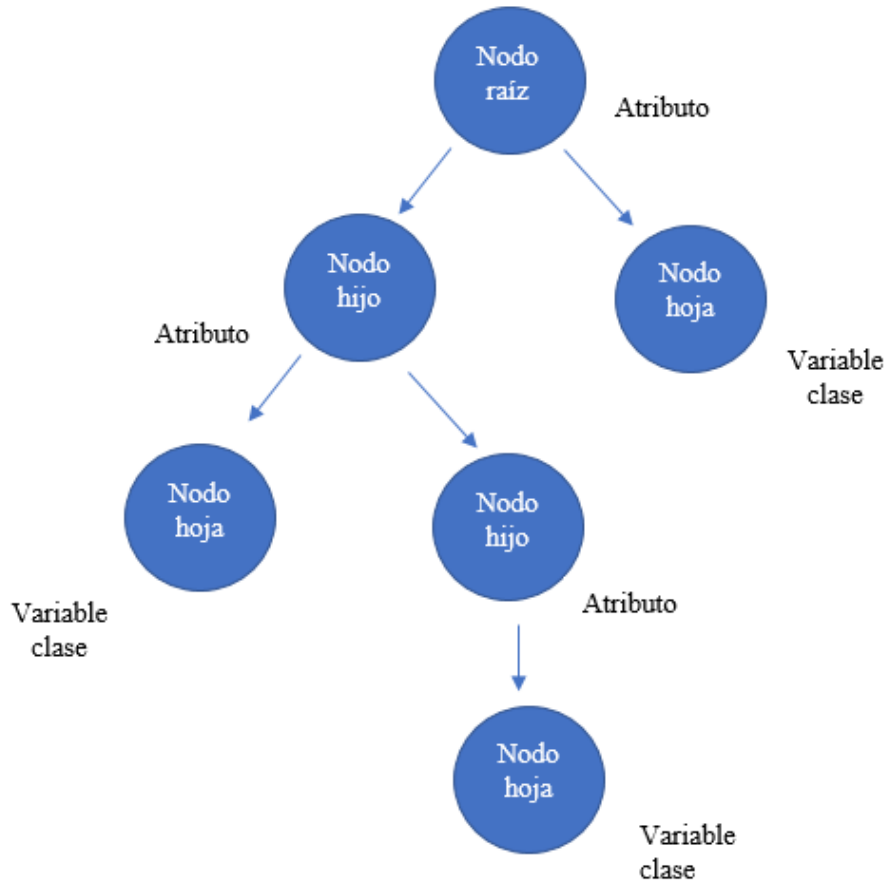
Fuente: (De La Hoz et al., 2019)

5.6. Árboles de decisión

Es un modelo de clasificación que se basa en un análisis predictivo, es decir a partir de datos con una estructura lógica se genera un árbol, este se representa por un conjunto de ramas, nodos y hojas.

En la **Figura *15***, se describe la estructura de un árbol de decisión en donde el nodo principal es un atributo que hace referencia al inicio de la clasificación, los nodos internos son las preguntas que se generan para cada variable de estudio. Cada respuesta de este nodo se denomina nodo hijo. Las ramas que se generan de cada nodo requieren de etiquetas con los valores posibles. Los nodos hojas representan la respuesta que coincide con cada variable clase. Es así que los árboles permiten identificar la relación que existe entre las variables de respuesta con la categoría a la que pertenece (Barrientos et al., 2019).

Figura 15. *Estructura de un árbol de decisión*



Fuente:(Barrientos et al., 2019)

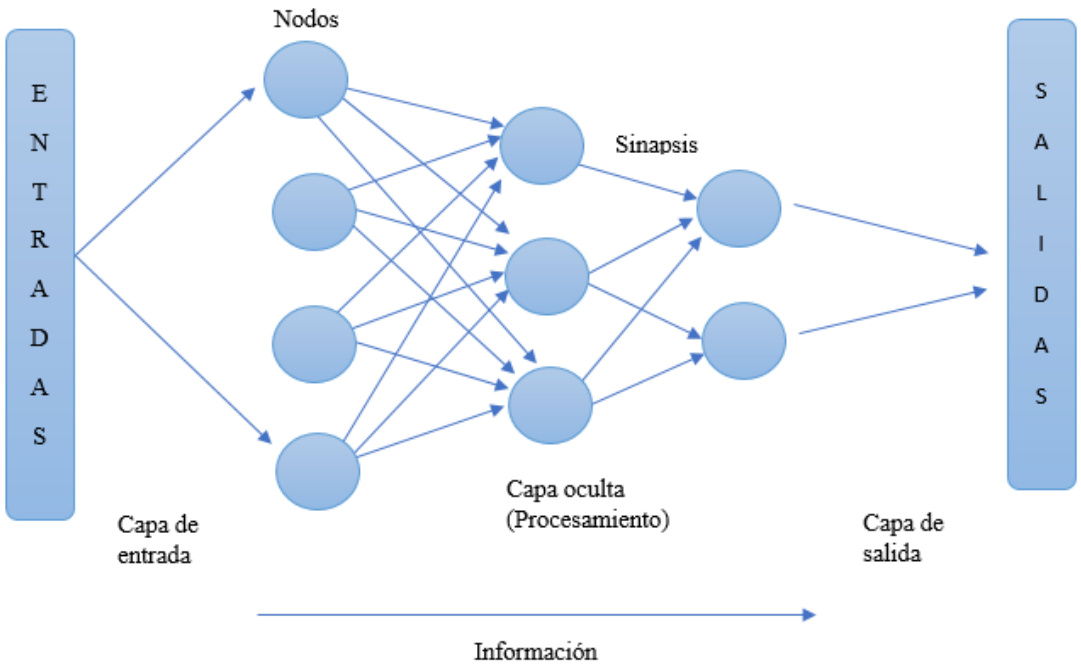
5.7. Redes Neuronales

Una red neuronal es considerada como un sistema que imita el comportamiento del cerebro humano, transformando entradas en saldas mediante un conjunto de neuronas. En una red neuronal los nodos se conectan por medio de sinapsis, adoptando el comportamiento de la red que se determinado por la estructura de conexiones sinápticas. Estas conexiones son direccionales, es decir, la información solamente puede propagarse en un único sentido, en la **Figura *16***, se muestra la arquitectura del modelo. En general las neuronas se suelen agrupar los daos en unidades estructurales que se denomina capas. El conjunto de una o más capas constituye la red neurona (Larranga & Moujahid, 2015).

Existe tres tipos de capas: la entrada la salida y la oculta. Una capa de entrada es aquella que recibe datos o señales de entrada para posterior procesar y enviar los resultados a una capa de salida. Capa oculta no tiene una conexión directa con el entorno, es decir que proporciona cierto grado de libertad a la red neuronal y debido a esto es capaz de

representar con mayor fiabilidad a ciertas características del entorno que se pretenda modelar (Larranga & Moujahid, 2015).

Figura 16. *Arquitectura de una red neuronal*



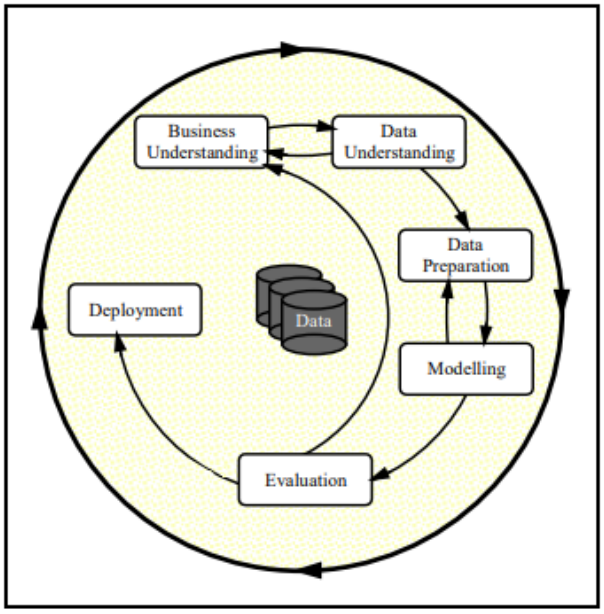
Fuente: (Larranga & Moujahid, 2015).

6. CAPÍTULO VI: CICLO DE VIDA DEL MODELO CRISP-DM

Para la implementación del modelo, se utilizó el entorno de desarrollo de Spyder que utiliza el lenguaje de programación de Phyton, este contiene librerías que permite programar modelos matemáticos y entrenar a los modelos de aprendizaje.

Para realizar las pruebas se utilizó la metodología de minería de datos CRISP-DM, en la **Figura *17*** se visualiza el ciclo de vida, este funciona por faces, primero se entiende el modelo de negocio, luego se entiende la data, se prepara, modela, evalúa y se despliega, en el caso que no cumpla con los datos que se espera, este vuelve a empezar hasta obtener el análisis predictivo adecuado.

Figura 17. *Ciclo de vida del modelo CRISP-DM*



Fuente: (Wirth & Hipp, 2020)

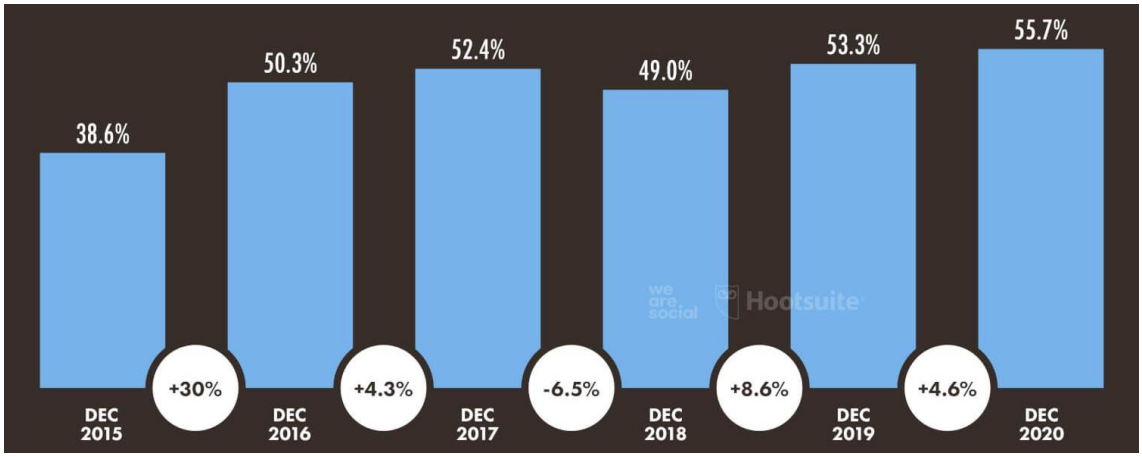
6.1. Entendimiento del negocio

Según SaasRank (2021) menciona que, entre los principales factores que promueven la industria de las apps, se tiene el aumento del número de usuarios de dispositivos digitales y el incremento de la velocidad del internet. Por lo que hasta el año 2021 se ha identificado a unos 4,32 mil millones de usuarios que utilizan dispositivos móviles, lo que representa al 92,6% de la población. Se estima que las personas utilizan un Smartphone en promedio 3 horas y 39 minutos al día, de ese lapso el 44% utilizan en redes sociales y comunicación,

el 26% en aplicaciones de video y entretenimiento, el 21% en otras aplicaciones, y el 9% en videojuegos (Shum, 2021).

Con estos datos se estima que el desarrollo de aplicaciones móviles irá en crecimiento, ya que con el transcurrir de los días los hábitos y necesidades de los usuarios cambian y se requiere de nuevas herramientas que faciliten la programación de aplicaciones con el objetivo de satisfacer los requerimientos.

Figura 18 Evolución del uso de aplicaciones móviles de los últimos 5 años



Fuente: (Shum, 2021).

6.2. Entendimiento de la data

En el entendimiento de la data se realizó una encuesta donde se estableció una restricción a los usuarios impidiendo que puedan terminar hasta llenar todas las preguntas, esto evitó tener datos faltantes. En el **Anexo 1** *se detallan las preguntas que componen la encuesta realizada. En la* **Figura 19** *se tiene 13 variables de entrada las cuales son categóricas y se identifica que las variables de salida son 5 clases que representan a los entornos de estudio. Los resultados que se obtuvo de cada pregunta se detallan en el* **Fase de Entendimiento**

Anexo *2*.

Figura 19. *Recolección de datos de la encuesta*

ID	¿Cuál de los	¿Qué nivel d	¿De qué m	¿A qué si	En qu	¿En qué nivel consid	Cuando terminas tu	¿En que nivel consid	Para e	¿Qué rango de tiem	¿Has	¿Qué aspecto ha sid	De los siguientes ca	¿Qué tipo de aplicac
1	Flutter	Principiante	Equipo	Ambos	Alto	Muy Importante	ambiente preparado	Muy Importante	Medio	6 a 12 meses	Si	Visual(Interfaz animacic	Entretenimiento	App móvil
2	Flutter	Medio	Equipo	Ambos	Alto	Nada Importante	Configuración manual	Medianamente Importa	Medio	3 a 6 meses	Si	Visual(Interfaz animacic	Empresas	Escritorio
3	Xamarin	Alto	Individual	Android	Alto	Nada Importante	ambiente preparado	Medianamente Importa	Bajo	6 a 12 meses	Si	Visual(Interfaz animacic	Administración	Sitio Web
4	Ionic	Medio	Equipo	Android	Medio	Medianamente Importa	Configuración manual	Medianamente Importa	Bajo	6 a 12 meses	Si	Seguridad	Negocio	Web y Escritorio
5	Cordova	Medio	Equipo	Android	Bajo	Medianamente Importa	ambiente preparado	Medianamente Importa	Medio	12 meses o mas	Si	Seguridad	Entretenimiento	Web y Escritorio

Dado que las variables son categóricas estas se requieren de una transformación que permita identificar la relación que tienen. Para esto se identificó la correlación existente

entre las variables de entrada con respecto a los entornos de estudio. Se obtuvo la correlación que se muestra en la **Figura *20***, en donde se visualiza que, existe una correspondencia entre los datos, lo cual permite determinar que el conjunto de datos puede ser modelado. Además, se puede identificar las variables que tienen mayor impacto para el modelo.

Otro aspecto importante de la correlación es que para poder definir este parámetro se basa en un rango establecido en el cual -1 corresponde a una correlación inversa perfecta y 1 corresponde a una correlación directa perfecta, en el caso de obtener un valor de 0 se dice que no existe correlación. También es necesario verificar que no se tenga correlación entre las variables de entrada, ya que esto ocasionaría que el modelo se sobreajuste.

Figura 20. *Correlación de datos*

	Experiencia	Modo	Ambiente	...	Aspectos	Campos	Entornos
Experiencia	1.000000	-0.176874	0.045244	...	0.015705	-0.048165	-0.117699
Modo	-0.176874	1.000000	-0.069452	...	-0.123743	0.028350	0.113586
Ambiente	0.045244	-0.069452	1.000000	...	0.063052	-0.015731	-0.064895
Madurez	-0.107429	0.028036	-0.242211	...	-0.003544	-0.053011	-0.009811
Presupuesto	-0.005181	0.058583	-0.176629	...	-0.058572	-0.115700	-0.013242
Tiempo	0.116669	-0.085058	0.159097	...	0.200492	-0.013895	0.006695
Aspectos	0.015705	-0.123743	0.063052	...	1.000000	-0.013719	-0.014700
Campos	-0.048165	0.028350	-0.015731	...	-0.013719	1.000000	0.031521
Entornos	-0.117699	0.113586	-0.064895	...	-0.014700	0.031521	1.000000

En este caso se identifica que las correlaciones son bajas entre las variables de entrada lo que permite suponer que se trata de variables independientes, entre ellas no hay correlación alta, lo cual es un buen indicio para determinar que las variables son independientes, no contra la variable de salida.

6.3. Preparación de la data

Para preparar los datos se requiere transformar las variables categóricas en numéricas para entrenar a la máquina, en este proceso es importante tener en cuenta que no se debe ingresar todo el conjunto de datos, debido a que el modelo puede presentar error al ingresar nuevos datos de entrada. Es por ello por lo que para crear un modelo que analice correctamente los datos se utiliza el concepto de validación cruzada, este permite diversificar la cantidad de datos para entrenamiento y prueba, esto dependerá de modelo que se esté desarrollando.

6.4. Variables de entrada

Las variables de entrada son los parámetros que se requieren para la formación de la data set a ser analizado, en la **Tabla *11*** se detallan las características de los entornos, y dependiendo de las opciones presentadas en el cuestionario se las pondera de manera numérica a través de rangos que dependerán del número de opciones de respuesta. En el **Anexo *4*** se presenta el dataset con las variables numéricas.

Tabla 11. *Escala de ponderación de cada una de las variables*

Características de los entornos (Variables)	Opciones	Valores que pueden tomar las variables
Nivel de Experiencia	Alto, Medio, Regular, Principiante	1,2,3,4,5
Modo de desarrollo	Individual, Equipo	1,2
Tipo de sistema operativo	Android, IOS, Ambos	1,2,3
Documentación	Nada Importante. Poco Importante, Medianamente Importante, Bastante Importante, Muy Importante	1,2,3,4,5
Despliegue de aplicación	Herramienta de Soporte, Configuración Manual	1,3
Estabilidad	Nada Importante. Poco Importante, Medianamente Importante, Bastante Importante, Muy Importante	1,2,3,4,5,
Presupuesto	No cuenta con presupuesto, < a 500 dólares, Entre 500 y 1500 dólares, Más de 1500 dólares	1,2,3,4
Rango de tiempo	3 meses o menos, 3 a 6 meses, 6 a 12 meses, 12 meses o más	1,2,3,4
Herramientas adicionales	Si, No	1,2
Aspectos para considerar en el desarrollo	Rendimiento, Visual (Interfaz animaciones), Consumo de batería, Seguridad	1,2,3,4
Campo de Aplicación	Administración, Educación, Empresas, Entretenimiento, Negocio, Salud, Social	1,2,3,4,5,6,7

Lenguaje actual de utilización	Lenguajes de Programación	1,2,3,4,5,6,7,8
Tipo de proyectos	Sitio Web, App móvil, Escritorio, Web y Móvil, Web y Escritorio, Móvil y Escritorio	1,2,3,4,5,6

6.5. Variables de Salida

Las variables de salida serán representadas por los frameworks, a los cuales se ha asignado una etiqueta para diferenciarlos, en la **¡Error! No se encuentra el origen de la referencia.** se muestra el valor que corresponde a cada uno.

Tabla 12. *Variables de salida*

Framework	Etiqueta
Flutter	1
ReactNative	2
Ionic	3
Xamarin	4
Cordova	5

6.6. Modelamiento

En primera instancia se seleccionó las técnicas de modelas a utilizar tales como: árbol de decisión, vecino cercano y red neuronal. El procedimiento para evaluar las condiciones en las cuales estos deben funcionar, se analiza la eficiencia que ha adquirido la máquina, este parámetro se encuentra estable en un 95%. Para la construcción del modelo se debe tomar en cuenta una validación cruzada, debido a que esto permitirá separar la data en un 75% para entrenamiento, y un 25 % para pruebas. Esto con el objetivo de que el modelo tome el valor de 1, que significa que la maquina ha tenido un sobreajuste, es decir tienen un buen rendimiento, pero no puede generalizar el entrenamiento para otros casos. También puede suceder lo contrario es decir la maquina no puede modelar y tampoco pueda generalizar.

Vecinos cercanos se basa en una variable que permite establecer puntos de referencia frente a otros, en este caso al generar una máquina de entrenamiento con knn se estableció la vecindad que existe entre las variables de entrada y las categorías de cada entorno.

Árboles de decisión se basa en un análisis descriptivo, en donde un nodo principal establece el inició de la clasificación y los nodos son decisiones que permiten predecir el resultado.

Redes neuronales emula un comportamiento similar al del cerebro, este utiliza todos los valores de entrada para realizar una suma ponderada de ellos la ponderación de cada una de las entradas viene dada por el peso que se le asigna a cada una de las conexiones de entradas es decir cada conexión que llega a nuestra neurona tendrá asociado un valor que servirá para definir con qué intensidad cada variable de entrada afecta a la neurona intuitivamente esto se suele representar como palancas que podemos subir o bajar para modificar positiva o negativamente el valor de nuestra suma como te imaginarás estos pesos son los parámetros de nuestro modelo y serán los valores que podremos ajustar para que nuestra red neuronal

6.7. Evaluación

Para el aprendizaje del modelo se dividió la data de prueba y la data de entrenamiento, esto se realizará de manera aleatoria, luego se declaró las variables que contienen los elementos que corresponde a entrenamiento y prueba. Se creó la máquina de aprendizaje con los tres tipos de algoritmos de clasificación, se parametrizó cada modelo y los resultados analizados en distintos casos fueron los siguientes:

En la **Tabla *13***, se muestra los parámetros versus las pruebas realizadas para el modelo clasificador de vecino cercano KNN.

Tabla 13. *Resultados algoritmo Vecino cercano KNN*

Vecino cercano KNN			
Parámetros	**P1**	**P2**	**P3**
Número de vecinos	8	9	9
Eficiencia	29%	34%	32%
Número de datos de entrenamiento	(X=288, y=97)	(X=288, y=97)	(X=288, y=97)
Tiempo de entrenamiento	1-2 seg	1-2 seg	1-2 seg
Score	0,4131944444	0,42361111111	0.41666666666

En la **Tabla *14***, se muestra los parámetros versus las pruebas realizadas para el modelo clasificador de árbol de decisión.

Tabla 14. *Resultados algoritmo Árbol de decisión*

Árboles de decisión			
Parámetros	**P1**	**P2**	**P3**
Profundidad del árbol	8	9	10
Eficiencia	63%	68%	78%
Número de datos de entrenamiento	(X=288, y=97)	(X=288, y=97)	(X=288, y=97)
Tiempo de entrenamiento	1-2 seg	1-2 seg	1-2 seg
Score	0.72916666666	0.77083333333	0.82986111111

En la **Tabla *15***, se muestra los parámetros versus las pruebas realizadas para el modelo clasificador de redes neuronales.

Tabla 15. *Resultados algoritmo Redes Neuronales*

Redes Neuronales			
Parámetros	**P1**	**P2**	**P3**
Iteraciones	10000	30000	9000
Tamaño de Capa oculta	800	10000	20000
Eficiencia	69%	74%	82%
Número de datos de entrenamiento	(X=288, y=97)	(X=288, y=97)	(X=288, y=97)
Tiempo de entrenamiento	5-7seg	15-20seg	40-60 seg
Score	0.77430555555	0.88541666666	0.90972222222

Luego de varias pruebas se identificó que el algoritmo de árboles de decisión y red neuronal pueden predecir el modelo, sin embargo, existe algunas diferencias entre estos que son: el tiempo y el rendimiento. La red neuronal analiza los datos en un lapso de 40 a 60 segundos, mientras que, en el caso del árbol de decisión, este es más rápido y puede predecir en menor tiempo. Al considerar el parámetro de eficiencia en el modelo se determinó que el aprendizaje de máquina de redes neuronales es el que mejor se ajusta, a pesar de que el tiempo de procesamiento es levemente más lento que los demás, tiene la capacidad de predecir adecuadamente.

Luego de analizar el modelo que mejor se adapta al caso de estudio, se procedió a realizar más pruebas para confirmar que el algoritmo cumplirá con futuras predicciones, los resultados obtenidos, es que al entrenar a nuevos datos utiliza 20000 iteraciones, con 900 capas ocultas y se consigue una eficiencia del 85%.

6.8. Despliegue

Las herramientas utilizadas para la implementación de sitio web se detallan a continuación:

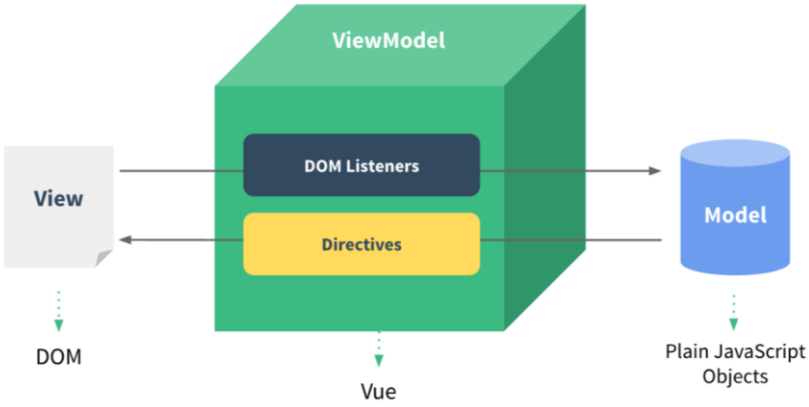
6.8.1. Frontend

Para el desarrollo frontend se utilizó el entorno vue.js, este es un marco de código abierto que se basa en HTML, CSS y Javascript. En la **Figura *21***, se muestra la arquitectura del entorno que se utiliza para la creación de interfaces de sitios web, aplicaciones móviles y escritorio, esto debido a que utiliza extensiones de html y la base de JavaScript que funcionan en conjunto, además utiliza una arquitectura MVC para desarrollar interfaces de usuarios.

Vue.js utiliza un sistema de renderización optimizado por un compilador que usualmente requiere de una configuración manual, es versátil debido a que, su ecosistema es incremental, es decir este puede escalar entre una biblioteca y un marco con todas sus funcionalidades.

La ventaja de este entorno es que optimiza el tiempo de desarrollo de manera que permite generar componentes reutilizables, disminuyendo la cantidad de código y agilizando el procesamiento de estos (Antamba, 2018).

Figura 21. *Arquitectura de Vue js*



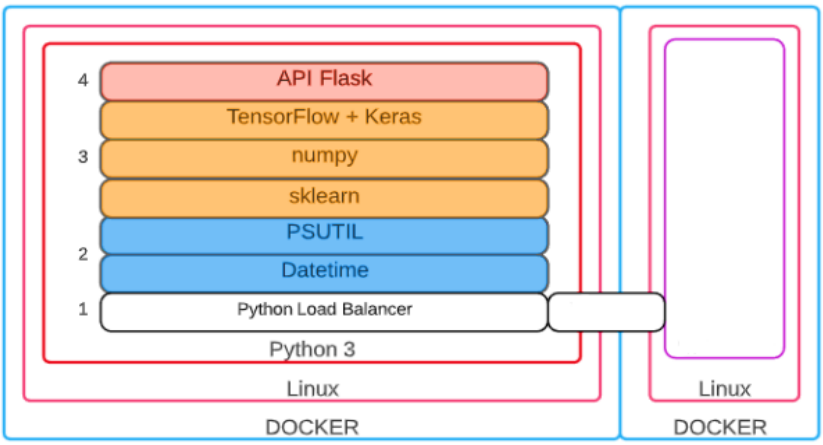
Fuente: (Antamba, 2018)

6.8.2. Backend

El backend del sitio que se utilizó es el micro framework Flask escrito en Python creado para desarrollar aplicaciones web dinámicas bajo el esquema MVC. Este permite utilizar herramientas o extensiones para programar funcionalidades más complejas en función de los requisitos que se desean desarrollar.

Flask por otro lado incluye un servidor web que permite ejecutar los cambios que se aplican al proyecto, es compatible con Python lo que permite crear controladores para manejar la data de la base de datos. En la **Figura 22**, se muestra la arquitectura del software del servidor.

Figura 22. Arquitectura de software del servidor



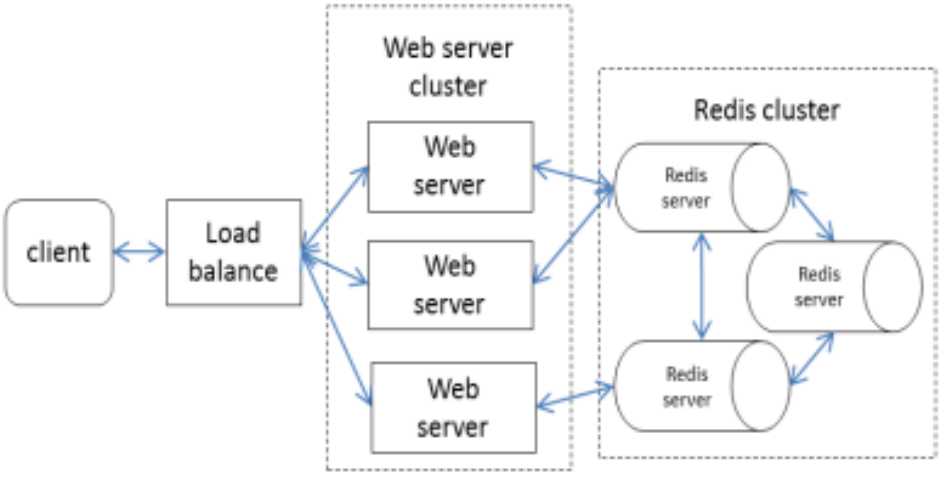
Fuente: (Izquierdo, 2020)

6.9. Base de datos

Se utilizó base de datos no relacionales (NoSql), las cuales están diseñadas para modelos de datos que requieren de flexibilidad, grandes cantidades de datos y baja latencia. En la **Figura *23***, se describe la arquitectura de la base de datos. En este caso se utilizó Redis, que es una base de datos dinámica de código abierto, permite manejar los datos en la memoria para un acceso óptimo, permite la creación de una infraestructura de datos para aplicaciones en tiempo real que requiere de un alto rendimiento y baja latencia.

La velocidad de Redis es ideal ya que permite almacenar datos en la caché consultar a la base de datos, llamadas a la API, cálculos complejos y mantiene el estado de la sesión.

Figura 23. Arquitectura de base de datos Redis



Fuente: (Li et al., 2018)

Pruebas adaptando el modelo entrenado con el sitio web

Para mostrar los resultados y el funcionamiento del modelo, se creó un sitio web que permite visualizar la información de cada entorno y a su vez el usuario tiene la posibilidad de dirigirse a los sitios disponibles por cada entorno para estudiarlo más a fondo.

El sitio web posee una sección que cuenta con una serie de preguntas relacionadas a los aspectos más relevantes al momento de empezar a desarrollar un proyecto, las respuestas servirán de entradas que serán insertadas a la base de datos, posterior a esto y a través de un controlador desarrollado en Flask se realiza la consulta al modelo que se entrenó previamente, este como resultado retorna el entorno que mejor se adapta. Para cada prueba se consideró el caso ideal, que permita validar el funcionamiento de cada uno de los entornos.

A continuación, se presenta las pruebas realizadas en el sitio web:

Prueba 1 Ingreso de la información para evaluar cada requerimiento.

En este caso se desea predecir el entorno de Flutter, para esto ingresaremos el caso ideal al sitio y visualizaremos cuál fue la respuesta. Las entradas se pueden visualizar en la **Tabla *16***.

Tabla 16. Datos de prueba framework Flutter

Entradas	Respuestas
Lenguajes de programación:	Dart
Nivel de experiencia	Principiante
Modo de desarrollo	Equipo
Sistema operativo	Ambos
Guía de soporte	Muy importante
Despliegue	Herramienta de soporte
Madurez del entorno	Bastante Importante
Presupuesto	Menos de 500 dólares
Tiempo	3 a 6 meses
Herramientas Extras	No
Aspectos para considerar	Visualización (Interfaz animaciones)
Campos de aplicación	Negocio

Resultado:

Prueba 2

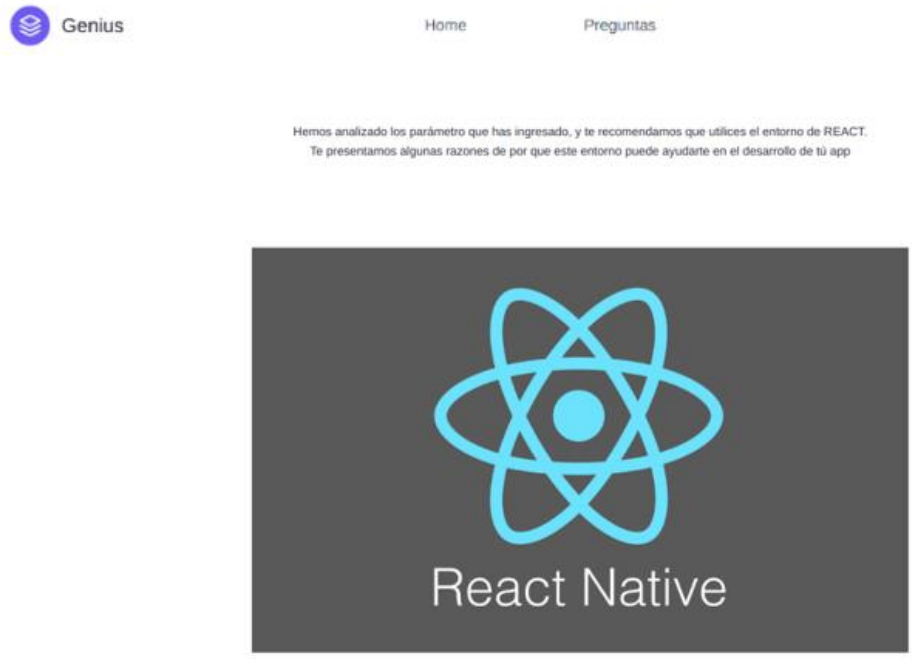
En este caso se desea predecir el entorno de React Native, para esto ingresaremos el caso ideal al sitio y corroboramos cuál fue la respuesta. Las entradas se pueden visualizar en la **Tabla *17***.

Tabla 17. Datos de prueba framework React Native

Entradas	Respuestas
Lenguajes de programación:	Java Script

Nivel de experiencia	Principiante
Modo de desarrollo	Individual
Sistema operativo	Ambos
Guía de soporte	Muy importante
Despliegue	Herramienta de soporte
Madurez del entorno	Bastante Importante
Presupuesto	No cuenta con presupuesto
Tiempo	12 meses o más
Herramientas Extras	No
Aspectos para considerar	Rendimiento
Campos de aplicación	Empresas

Resultado:



Prueba 3

En este caso se desea predecir el entorno de Ionic, para esto ingresaremos el caso ideal al sitio y corroboramos cuál fue la respuesta. Las entradas se pueden visualizar en la Tabla 18.

Tabla 18

Tabla 18. Datos de prueba framework Ionic

Entradas	Respuestas
Lenguajes de programación:	Java Script
Nivel de experiencia	Principiante
Modo de desarrollo	Individual
Sistema operativo	Ambos
Guía de soporte	Bastante Importante
Despliegue	Herramienta de soporte
Madurez del entorno	Bastante Importante
Presupuesto	Más de 1500 dólares
Tiempo	6 a 12 meses
Herramientas Extras	No
Aspectos para considerar	Rendimiento
Campos de aplicación	Empresas

Resultado:

Prueba 4

En este caso se desea predecir el entorno de Xamarin, para esto ingresaremos el caso ideal al sitio y corroboramos cuál fue la respuesta. Las entradas se pueden visualizar en la **Tabla *19*.**

Tabla 19. Datos de prueba framework Xamarin

Entradas	Respuestas
Lenguajes de programación:	C#
Nivel de experiencia	Principiante
Modo de desarrollo	Individual
Sistema operativo	Ambos
Guía de soporte	Bastante Importante
Despliegue	Herramienta de soporte
Madurez del entorno	Bastante Importante
Presupuesto	Menos de 500 dólares
Tiempo	12 meses o más
Herramientas Extras	Si
Aspectos para considerar	Consumo de batería
Campos de aplicación	Administración

Resultado:

Prueba 5

En este caso se desea predecir el entorno de Cordova, para esto ingresaremos el caso ideal al sitio y corroboramos cuál fue la respuesta. Las entradas se pueden visualizar en la **Tabla *20*.**

Tabla 20. Datos de prueba framework Cordova

Entradas	Respuestas
Lenguajes de programación:	Java Script

Nivel de experiencia	Principiante
Modo de desarrollo	Equipo
Sistema operativo	Ambos
Guía de soporte	Bastante Importante
Despliegue	Configuración Manual
Madurez del entorno	Muy Importante
Presupuesto	Entre 500 y 1500 dólares
Tiempo	6 a 12 meses
Herramientas Extras	No
Aspectos para considerar	Funcionamiento fuera de línea (Off line)
Campos de aplicación	Salud

Resultado:

7. CAPITULO VII: CONCLUSIONES Y RECOMENDACIONES

7.1. Conclusiones

Se identificó que cada entorno de desarrollo no posee un campo de aplicación definido, de manera que estos pueden ser utilizados en cualquier rama. En cuanto a los resultados obtenidos se concluye que el modelo se adapta a los requerimientos que se planteó para este trabajo.

Durante la investigación desarrollada se logró identificar qué aspectos diferencian a cada entorno. Flutter se ha caracterizado por ser un entorno que permite desarrollar proyectos en corto tiempo, con una interfaz optimizada, y disminución de costes en la implementación. React Native por otro lado se enfoca en el rendimiento y funcionalidades nativas, por su lenguaje de programación este permite migrar aplicaciones web hacia aplicaciones móviles, lo que representa una disminución en tiempo en el desarrollo y eficiencia en cuanto a funcionalidades.

Ionic a más de ofrecer la creación de apps híbridas, también permite desplegar desde el mismo entorno aplicaciones para los sistemas operativos tanto Android como iOS, este como tal provee de servicios empresariales, que significa que se requiera de un presupuesto alto dependiendo del enfoque y necesidades de una empresa. Xamarin al utilizar el lenguaje de programación C# se encuentra adherido a .Net, permitiendo utilizar funcionalidades de servidores y a su vez ofrece servicios en la nube de Microsoft Azure, esto incrementaría un costo adicional en el presupuesto, pero tiene la ventaja de que mediante plugins se puede adaptar a las funcionalidades de un proyecto.

Cordova aprovecha el rendimiento nativo de los dispositivos completamente a través de Apis, considerando aspectos como: estado de la batería, cámara, geolocalización, información de red, vibración, entre otros. Este entorno a más de enfocarse en el funcionamiento del dispositivo también provee soporte de escenarios fuera de línea, lo que hace de este framework una gran diferencia frente a los demás y proporciona una alternativa para aquellos proyectos que tienen como finalidad proveer un mejor servicio a sus clientes y aprovechar al máximo el aplicativo.

El modelo escogido es redes neuronales que cumple con el objetivo de predecir que entorno tiene más relación con los requerimientos que se necesite en un proyecto de desarrollo de aplicaciones móviles, y que además está en la capacidad de pronosticar con nuevos datos de entrada y mostrar un resultado favorable.

La inteligencia artificial para el desarrollo de este proyecto permitió analizar patrones, que se encuentran en los datos, ya que a simple vista no se podía encontrar algún tipo de análisis. La IA al utilizar modelos matemáticos y estadísticos nos proporciona una manera amplia de manejar la información y dependiendo del enfoque que se tenga en el estudio, podemos obtener respuestas que pueden ser posibles soluciones para la toma de decisiones.

En cuanto al análisis que se realizó a través de encuestas, se identificó que entre los usuarios la mayor parte son desarrolladores principiantes, con este dato se puede corroborar que efectivamente existe una dificultad al momento de implementar un proyecto, siendo la experiencia un factor que influye en la decisión para poder definir que herramienta se adecúa a las necesidades de un proyecto.

Se estableció que la mejor forma de presentar resultados fue a través en un sitio web que funciona con inteligencia artificial. Al presentar un cuestionario en donde el usuario debe seleccionar la respuesta según su necesidad, con lo cual se concluye que es un sitio interactivo y fácil de utilizar.

7.2. Recomendaciones

En la implementación de un proyecto, se deben considerar varios aspectos como: el presupuesto necesario, el tiempo que tomará implementar los requisitos del proyecto, la experiencia o habilidad para adaptarse a un nuevo entorno, la madurez, etc. Esto con la finalidad de definir correctamente las fases de desarrollo y evitar retrasos o malas decisiones al iniciar un proyecto.

Para la recolección de datos es importante tomar en cuenta el tipo de análisis que se desea realizar, ya que de esto depende la manera en cómo se deben formular el cuestionario para la recolección de información y también para definir las variables de respuesta que permitan obtener datos que aporten al tema de investigación. De esta manera será más claro ponderar a las respuestas y posterior proceder a la depuración y limpieza de la data.

Es así que esto permitirá que al momento de probar los datos exista una correlación entre las variables de entrada y salida, y a partir de esa información se pueda modelar adecuadamente.

Con respecto al dataset se debe comprobar que no contenga espacios y el formato de los valores sea numérico, debido a que, al momento de cargar el archivo en el modelo, este puede presentar errores de toquenización.

8. BIBLIOGRAFÍA

Antamba, A. (2018). *Desarrollo Del Sistema Web Para La Gestión Académica De La Unidad Educativa "Modesto a. Peñaherrera". Utilizando Las Herramientas Vue.Js Y Spring Framework.* 1–99. https://www.researchgate.net/profile/Pedro-Larranaga/publication/268291232_Tema_8_Redes_Neuronales/links/55b7b5c408ae9289a08c0c68/Tema-8-Redes-Neuronales.pdf

Barrientos, R., Cruz, N., Acosta, H., Rabatte, I., & Gogeascoechea, M. (2019). Árboles de decisión como herramienta en el diagnóstico Médico. *Articulo Original*, 20–24. https://scielo.conicyt.cl/pdf/infotec/v30n1/0718-0764-infotec-30-01-247.pdf

Cárdenas Villavicencio, O. E., Zea Ordóñez, M. P., Valarezo Pardo, M. R., & Ramón Ramón, R. A. (2021). Comparativa de tendencias de desarrollo de software móvil. *3C TIC: Cuadernos de Desarrollo Aplicados a Las TIC*, *10*(1), 123–147. https://doi.org/10.17993/3ctic.2021.101.123-147

Cordova. (2021). *Cordova*. https://cordova.apache.org/

Daisy, I., & Imbaquingo, E. (2018). *ESTUDIO COMPARATIVO DE LOS FRAMEWORKS IONIC Y REACT NATIVE" APLICACIÓN MÓVIL DE PEDIDOS A DOMICILIO BASADA EN LA NORMA ISO*. https://riunet.upv.es/handle/10251/128486

De La Hoz, E., De La Hoz, E., & Fontalvo, T. (2019). Methodology of Machine Learning for the classification and Prediction of users in Virtual Education Environments. *Informacion Tecnologica*, *30*(1), 247–254. https://doi.org/10.4067/S0718-07642019000100247

Espinoza, J. (2020). *ANÁLISIS DE LOS FRAMEWORKS JAVASCRIPT NATIVO Y ANGULAR EN LA INCIDENCIA DEL TIEMPO DE RESPUESTA EN UNA WEB MVC EN EL SECTOR COMERCIAL* [Universidad Privada del Norte]. https://repositorio.upn.edu.pe/bitstream/handle/11537/24254/Espinoza Pereda%2C Juan Carlos.pdf?sequence=1&isAllowed=y

Flutter. (2021). *Flutter*. https://flutter.dev/?gclid=Cj0KCQjw4eaJBhDMARIsANhrQAA9GznRplhodU85j1dtj5YHisANdK4pi5t79rb8YuS2X9v49Zb9EQgaAo0-EALw_wcB&gclsrc=aw.ds

Gutiérrez, L. (2019). *BIG DATA aplicado al análisis de aplicaciones.* http://calderon.cud.uvigo.es/bitstream/handle/123456789/320/20191108 Resumen BIG DATA APLICADO AL ANALISIS DE APLICACIONES ANDROID GUTIERREZ nueva plantilla.pdf?sequence=1&isAllowed=y

Ionic. (2020). *Ionic*. https://ionicframework.com/

Izquierdo, F. (2020). *Diseño de una arquitectura de redes neuronales convolucionales distribuidas Diego Fernando Izquierdo Dussan Mateo Alfonso Puerto Rodríguez* [Universidad Distrital Francisco José de Caldas]. https://repository.udistrital.edu.co/bitstream/handle/11349/27971/PuertoRodriguezMateoAlfonso2020.pdf?sequence=1&isAllowed=y

Larranga, P., & Moujahid, A. (2015). *Redes Neuronales*. *July*, 1–19. https://www.researchgate.net/profile/Pedro-Larranaga/publication/268291232_Tema_8_Redes_Neuronales/links/55b7b5c408ae9289a08c0c68/Tema-8-Redes-Neuronales.pdf

Li, S., Jiang, H., & Shi, M. (2018). Redis-based web server cluster session maintaining technology. *ICNC-FSKD 2017 - 13th International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery*, 3065–3069. https://doi.org/10.1109/FSKD.2017.8393274

Oussous, A., Benjelloun, F. Z., Ait Lahcen, A., & Belfkih, S. (2018). Big Data technologies: A survey. *Journal of King Saud University - Computer and Information Sciences*, *30*(4), 431–448. https://doi.org/10.1016/j.jksuci.2017.06.001

ReactNative. (2021a). *Documentación React Native*. https://reactnative.dev/

ReactNative. (2021b). *React Native*. https://reactnative.dev/

Rouhiainen, L. (2018). Inteligencia artificial. *Alienta Editorial*, 352. https://planetadelibrosar0.cdnstatics.com/libros_contenido_extra/40/39307_Inteligencia_artificial.pdf

Ruiz, A., Arciniegas, J. L., & Giraldo, W. J. (2018). Caracterización de marcos de desarrollo de la interfaz de usuario para sistemas interactivos basados en distribución de contenido de video. *Ingeniare. Revista Chilena de Ingeniería*, *26*(2), 339–353. https://doi.org/10.4067/s0718-33052018000200339

SaasRank. (2021). *Datos y estadísticas de SaaS y mercado de las apps en 2021*. https://saasrank.es/datos-estadisticas-de-saas-mercado-apps-en-2021/

Shah, K., Sinha, H., & Mishra, P. (2019). Analysis of Cross-Platform Mobile App Development Tools. *2019 IEEE 5th International Conference for Convergence in Technology (I2CT)*, 1–7.

Shum, Y. M. (2021). *Situación Global Mobile o Móvil 2021*. https://yiminshum.com/mobile-movil-mundo-2021/

Sultan, M. (2017). *Angular y los marcos de tendencias de los dispositivos móvilesy tecnologías de plataforma basadas en la web Análisis comparativoitle*. 928–936.

Terfehr, N. (2022). *Statista*. https://es.statista.com/

Topón, J. (2020). *Validación del Framework Integrado para el Desarrollo de Aplicaciones Móviles IFMAD a través del desarrollo de la app Mi Agenda CDI* [Universidad de las Fuerzas Armadas Espe]. http://dx.doi.org/10.1016/j.ndteint.2014.07.001%0Ahttps://doi.org/10.1016/j.ndteint.2017.12.003%0Ahttp://dx.doi.org/10.1016/j.matdes.2017.02.024

Valdivia, J. (2016). *Modelo De Procesos Para El Desarrollo Del*. 187–208.

Valdiviezo, P. (2019). *Sistema recomendador híbrido basado en modelos probabilísticos* [Universidad Politécnica de Madrid]. https://oa.upm.es/57250/1/PRISCILA_MARISELA_VALDIVIEZO_DIAZ_2.pdf

Vázquez, V. (2019). Desarrollo de aplicaciones móviles multiplataforma con Flutter. *Universidad de Almería*, 72. http://repositorio.ual.es/bitstream/handle/10835/8010/TFG_VAZQUEZ RODRIGUEZ, VICTOR.pdf?sequence=1

Wirth, R., & Hipp, J. (2020). CRISP-DM: towards a standard process model for data mining. Proceedings of the Fourth International Conference on the Practical Application of Knowledge Discovery and Data Mining, 29-39. *Proceedings of the Fourth International Conference on the Practical Application of Knowledge Discovery and Data Mining*, *24959*, 29–39. https://www.researchgate.net/publication/239585378_CRISP-DM_Towards_a_standard_process_model_for_data_mining

Xamarin. (2021). *Xamarin*. https://dotnet.microsoft.com/apps/xamarin

Yaguapaz, L. (2018). *ESTUDIO DEL FRAMEWORK IONIC 2 PARA EL DESARROLLO DE APLICACIONES MÓVILES HIBRIDAS*. UNIVERSIDAD TÉCNICA DEL NORTE.

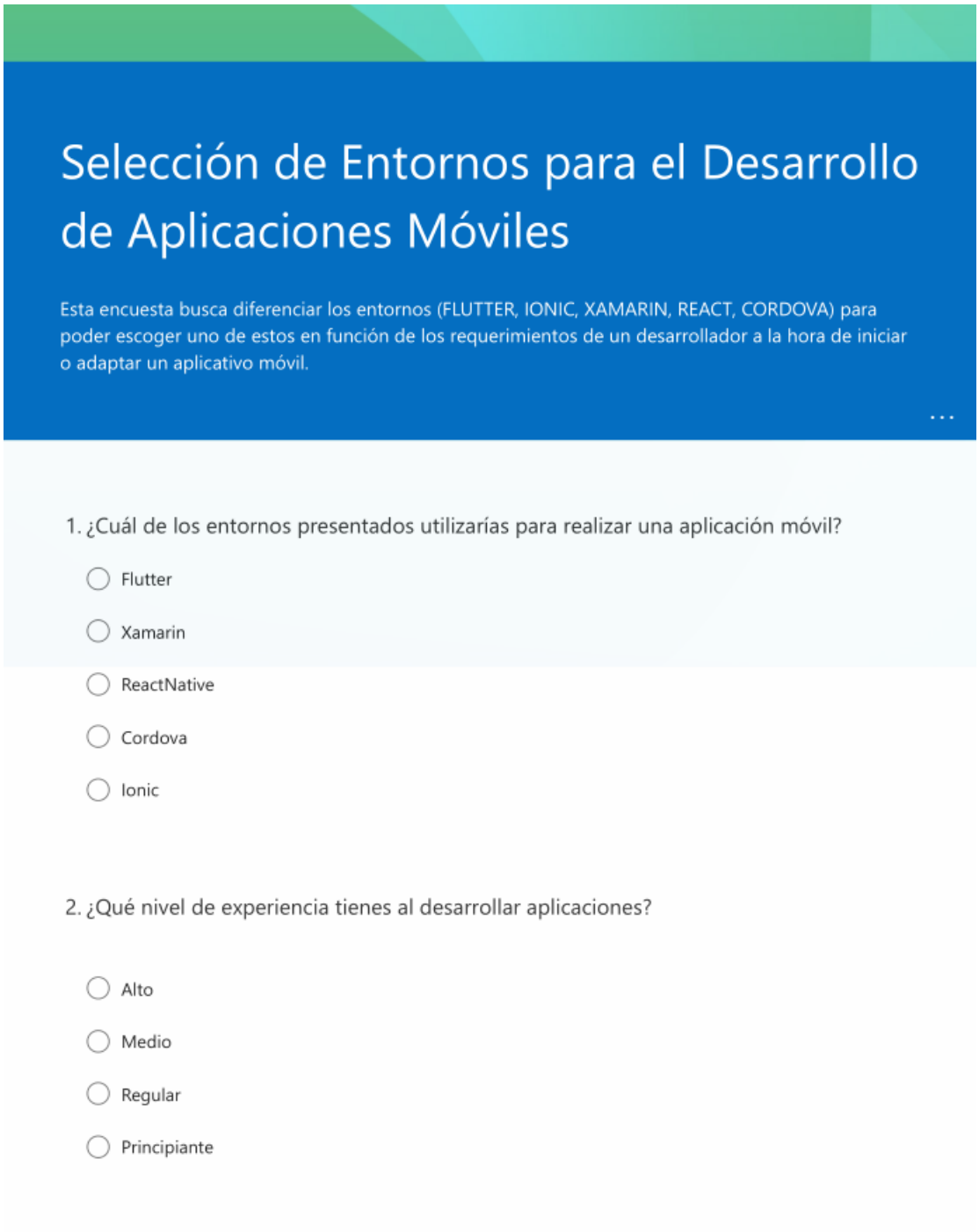
Zatarain, J. (2019). *Implementación del Framework React-Native en aplicaciones Nativas para la optimización de procesos* [Universidad Politécnica de Sinaloa]. http://repositorio.upsin.edu.mx/Fragmentos/tesinas/062016030035ZatarainGutierrezJulioCesar8143.pdf

Zuñiga, L. (2020). *Desarrollo de aplicaciones web utilizando Angular como framework* [Universidad Politécnica de Sinaloa]. http://repositorio.upsin.edu.mx/Fragmentos/tesinas/A021ZUNIGAVAZQUEZLUISRAMON9312.pdf

ANEXOS

Fase entendimiento del Negocio

Anexo 1. Encuesta para recolección de información



Selección de Entornos para el Desarrollo de Aplicaciones Móviles

Esta encuesta busca diferenciar los entornos (FLUTTER, IONIC, XAMARIN, REACT, CORDOVA) para poder escoger uno de estos en función de los requerimientos de un desarrollador a la hora de iniciar o adaptar un aplicativo móvil.

...

1. ¿Cuál de los entornos presentados utilizarías para realizar una aplicación móvil?

- ◯ Flutter
- ◯ Xamarin
- ◯ ReactNative
- ◯ Cordova
- ◯ Ionic

2. ¿Qué nivel de experiencia tienes al desarrollar aplicaciones?

- ◯ Alto
- ◯ Medio
- ◯ Regular
- ◯ Principiante

3. ¿De qué manera prefieres desarrollar aplicaciones móviles?

- ◯ Individual
- ◯ Equipo

4. ¿A qué sistema operativo va dirigida tu aplicación móvil?

- ◯ Android
- ◯ iOS
- ◯ Ambos

5. ¿En qué nivel considerarías que la documentación de un entorno es importante?

- ◯ Nada Importante
- ◯ Poco Importante
- ◯ Medianamente Importante
- ◯ Bastante Importante
- ◯ Muy Importante

6. Al terminar tu aplicación, ¿Prefiere utilizar una herramienta de soporte para el despliegue o prefiere configurarlo todo manualmente?

- ◯ Herramienta de Soporte
- ◯ Configuración Manual

7. ¿En que nivel considera importante que utilizar un entorno con mayor tiempo en el mercado le ayudará a desarrollar su aplicación?

- ◯ Nada Importante
- ◯ Poco Importante
- ◯ Medianamente Importante
- ◯ Bastante Importante
- ◯ Muy Importante

8. Para el desarrollo de sus aplicaciones, ¿Con qué presupuesto cuenta actualmente?

- ◯ No cuenta con presupuesto
- ◯ < a 500 dólares
- ◯ Entre 500 y 1500 dólares
- ◯ Más de 1500 dólares

9. ¿En qué rango de tiempo espera desarrollar una aplicación móvil?

- ◯ 3 meses o menos
- ◯ 3 a 6 meses
- ◯ 6 a 12 meses
- ◯ 12 meses o más

10. ¿Estaría dispuesto a pagar por herramientas extras para desarrollar su app?

- ◯ Si
- ◯ No

11. ¿Qué aspecto considera es el más importante a la hora de desarrollar una aplicación?

- ◯ Rendimiento
- ◯ Visual (Interfaz animaciones)
- ◯ Consumo de batería
- ◯ Seguridad

12. De los siguientes campos ¿cuál sé asocial al desarrollo de su app?

- ◯ Administración
- ◯ Educación
- ◯ Empresas
- ◯ Entretenimiento
- ◯ Negocio
- ◯ Salud
- ◯ Social

13. ¿Qué lenguaje de programación estás utilizando actualmente?

Esta pregunta es opcional para tener un conocimiento de que lenguaje en general es el más utilizado

Escriba su respuesta

14.

Qué tipo de proyecto te encuentras desarrollando actualmente?

- ◯ Sitio Web
- ◯ App móvil
- ◯ Escritorio
- ◯ Web y Móvil
- ◯ Web y Escritorio
- ◯ Movil y Escritorio

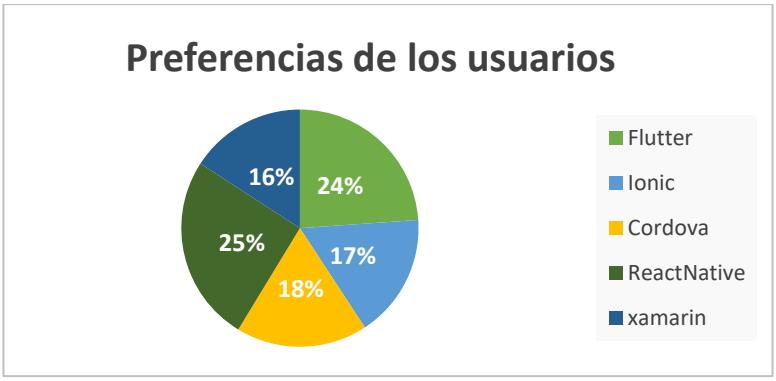
Fase de Entendimiento

Anexo 2. Resultados de la encuesta

1. **¿Cuál de los entornos presentados utilizarías para realizar una aplicación móvil?**

En cuanto a la preferencia de entornos para el desarrollo de aplicaciones móviles, se muestra en la **Figura *24*** los resultados: Flutter obtuvo la calificación de 92 usuarios que lo calificaron como preferente, esto representa al 24%; Ionic fue calificado como de su preferencia por 65 usuarios, lo que representa al 17%; Cordova con 69 usuarios lo que representa al 18%; React Native con 98 usuarios, lo que representa al 25%; Xamarin 61 usuarios, lo que representa al 16%. Con esto se puede establecer que el entorno más utilizado es React Native.

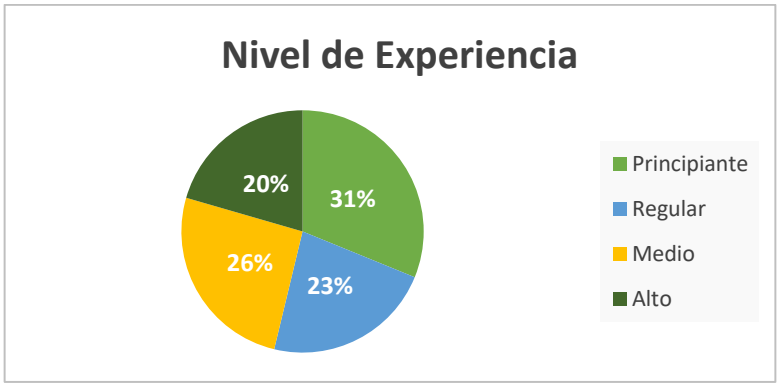
Figura 24. *Preferencia de los usuarios*



2. **¿Qué nivel de experiencia tienes al desarrollar aplicaciones?**

En la **Figura *25***, se muestran los resultados respecto al nivel de experiencia que han adquirido los desarrolladores se obtuvo que: a principiante lo escogieron 120 usuarios, lo que representa al 31%; a regular 87 usuarios, lo que representa al 23%; a medio 99 usuarios, lo que representa al 26%; a alto 79 usuarios lo que representa al 20%. Con lo que se concluye que el nivel principiante es el que predomina entre los usuarios.

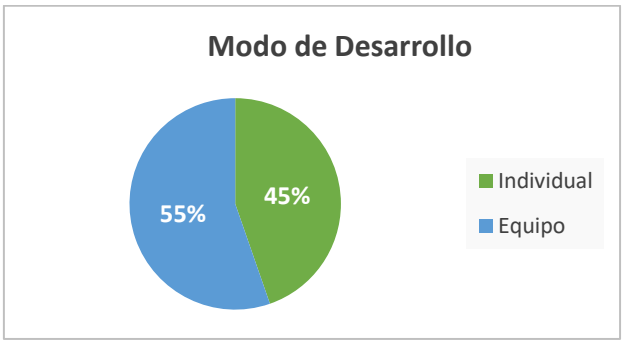
Figura 25. Nivel de Experiencia



3. ¿De qué manera prefieres desarrollar aplicaciones móviles?

En la manera de desarrollar las aplicaciones móviles, se presenta en la **Figura *26***, y se muestra que: individualmente son 172 usuarios, que representa al 45%; en equipo 213 usuarios, lo que representa al 55%. Como resultado de este análisis de concluye que la mayoría de los desarrolladores prefieren trabajar en equipo que brinden apoyo.

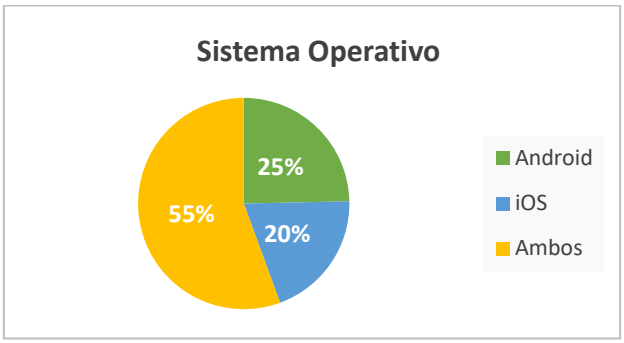
Figura 26. *Modo de Desarrollo*



4. ¿A qué sistema operativo va dirigida tu aplicación móvil?

Respecto a la cantidad de proyectos que han desarrollado para los sistemas operativos, en la **Figura *27*** se muestra los resultados y se observa que para: Android seleccionaron 95 usuario, lo que representa al 25%; iOS 76 usuarios, lo que representa al 20%; y ambos 214 usuarios, lo que representa al 55%. Con lo que se concluye que los usuarios desarrollan las aplicaciones móviles para ambos sistemas operativos.

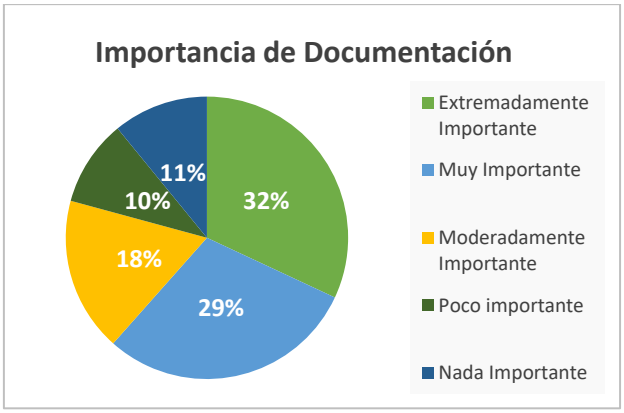
Figura 27. *Sistema Operativo*



5. **¿En qué nivel considerarías que la documentación de un entorno es importante?**

Respecto a la importancia de tener una guía de documentación para entender mejor al entorno, y de acuerdo con lo que se muestra en la **Figura *28*** se obtuvo que: extremadamente importante seleccionaron 123 usuarios, lo que representa al 25%; muy importante 114 usuarios, lo que representa al 29%; moderadamente importante 68 usuarios, lo que representa al 18%; poco importante 38 usuarios, lo que representa al 10%; nada importante 42 usuarios, lo que representa al 11%. Con lo que se concluye que la importancia de documentación es muy importante para guiar un proyecto.

Figura 28. *Importancia de la documentación*

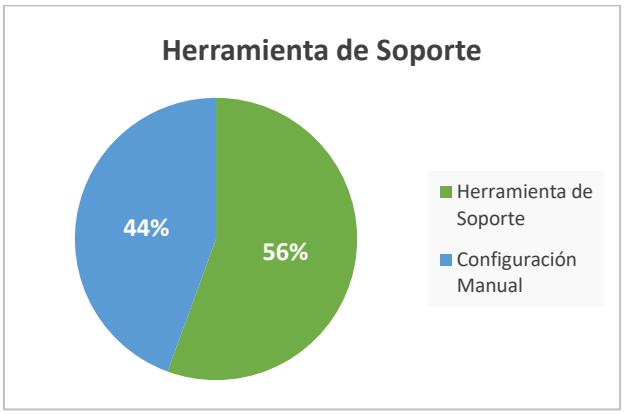


6. **Al terminar tu aplicación, ¿Prefiere utilizar una herramienta de soporte para el despliegue o prefiere configurarlo todo manualmente?**

Respecto a la utilización de herramientas extras para implantar los proyectos realizados, según se muestra en la **Figura *29***, se obtuvo que: los que prefieren herramienta de soporte son 214 usuarios, lo que representa al 56%; y los que prefieren

configuración manual son 171 usuarios, lo que representa al 44%. Con los resultados se define que los desarrolladores prefieren que se cuente con herramientas de soporte para dar funcionalidad a la aplicación

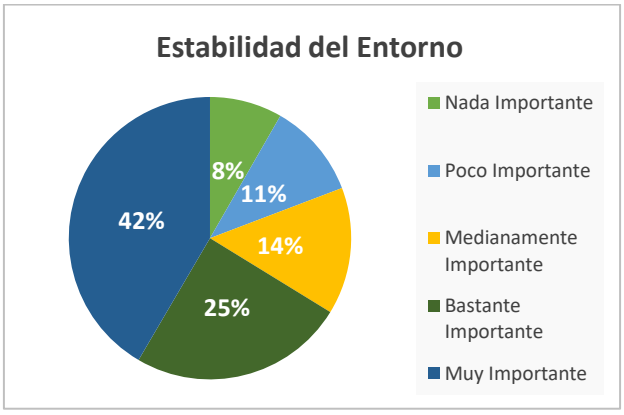
Figura 29. *Herramientas de Soporte*



7. **¿En qué nivel considera importante que utilizar un entorno con mayor tiempo en el mercado le ayudará a desarrollar su aplicación?**

Respecto a la importancia de utilizar un entorno que ya ha sido probado y se pone a disposición de los desarrolladores para implementación, según la **Figura *30*** se muestra que el nivel de: muy importante los escogieron 160 usuarios, lo que representa al 42%; bastante importante lo escogieron 95 usuarios, lo que representa al 25%; medianamente importante lo escogieron 56 usuarios, lo que representa al 14%; poco importante lo escogieron 42 usuarios, lo que representa al 11%; y nada importante lo escogieron 32 usuarios, lo que representa al 8%. Con lo que se concluye que los usuarios expresan que es muy importante que el entorno tenga mayor tiempo en el mercado, dando mayor confiabilidad.

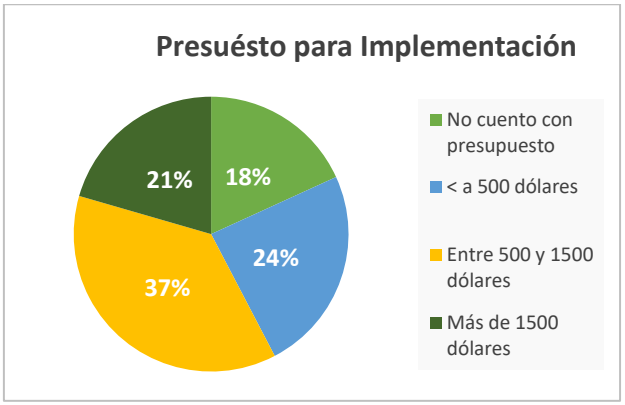
Figura 30. *Estabilidad del Entorno*



8. **Para el desarrollo de sus aplicaciones, ¿Con qué presupuesto cuenta actualmente?**

Respecto al rango de presupuesto que manejan durante el desarrollo, y según lo que se muestra en la **Figura *31***, se obtuvo que: no contar con presupuesto son 70 usuarios, lo que representa al 18%; menor a 500 dólares son 93 usuarios, lo que representa al 24%; entre 500 y 1500 dólares son 143 usuarios, lo que representa al 37%; más de 1500 dólares son 79 usuarios, lo que representa al 21%. Con esto se concluye que la mayoría cuenta con un presupuesto de entre $500 y $1500 para la ejecución de la aplicación móvil.

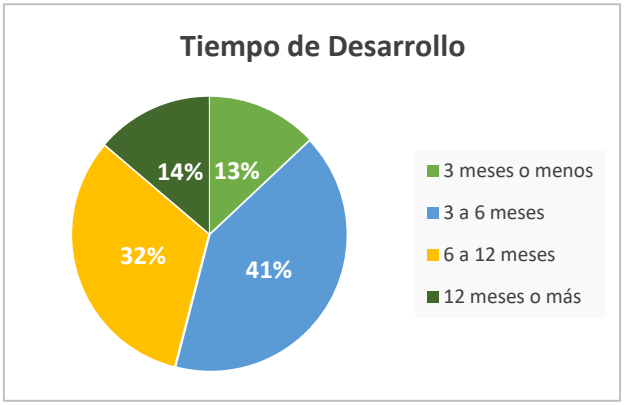
Figura 31. *Presupuesto para la implementación*



9. **¿En qué rango de tiempo espera desarrollar una aplicación móvil?**

Respecto al tiempo disponible para el desarrollo, se muestra en la **Figura *32*** que: 50 usuarios indican que son 3 meses o menos, lo que representa al 13%; 158 usuarios requieren de 3 a 6 meses, lo que representa al 41%; 124 usuarios requieren de 6 a 12 meses, lo que representa al 32%; 53 usuarios requieren 12 meses o más, lo que representa al 14%. Con este resultado se concluye que el tiempo utilizado con mayor frecuencia es de 6 a 12 meses para el desarrollo de la aplicación.

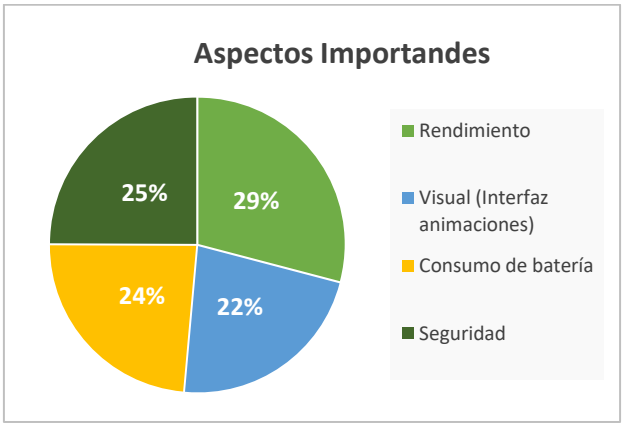
Figura 32. *Tiempo de Desarrollo*



10. ¿Qué aspecto considera es el más importante a la hora de desarrollar una aplicación?

Respecto a los aspectos a tomar en cuenta para el desarrollo de una aplicación, según la **Figura 33**, se visualiza que: el rendimiento lo han escogido 112 usuarios, lo que representa al 29%; la parte visual es decir la interfaz la han escogido 86 usuarios, lo que representa al 22%; el consumo de batería lo han escogido 91 usuarios, lo que representa al 24%; y la seguridad la han escogido 96 usuarios, lo que representa al 25%. Con estos resultados el aspecto que más destaca en los usuarios es el rendimiento.

Figura 33. *Características importantes*

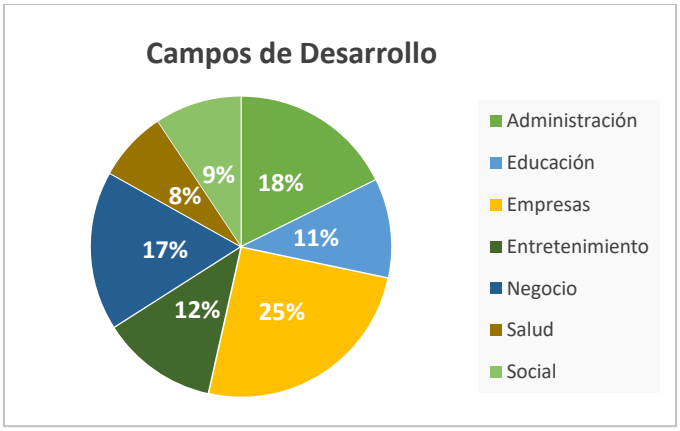


11. De los siguientes campos ¿cuál sé asocial al desarrollo de su app?

Respecto a los campos a los cuales se orientan las aplicaciones, en la **Figura *34*** se muestra que para: Administración han escogido 68 usuarios, lo que representa al 18%;

Educación han escogido 41 usuarios, lo que representa al 11%; Empresas han escogido 97 usuarios, lo que representa al 25%; Entretenimiento han escogido 48 usuarios, lo que representa al 12%; Negocio han escogido 66 usuarios, lo que representa al 17%; Salud han escogido 29 usuarios, lo que representa al 8%; Social lo han escogido 36 usuarios, lo que representa al 9%. Con estos resultados se evidencia que una aplicación móvil, puede ser desarrollada para distintos sectores.

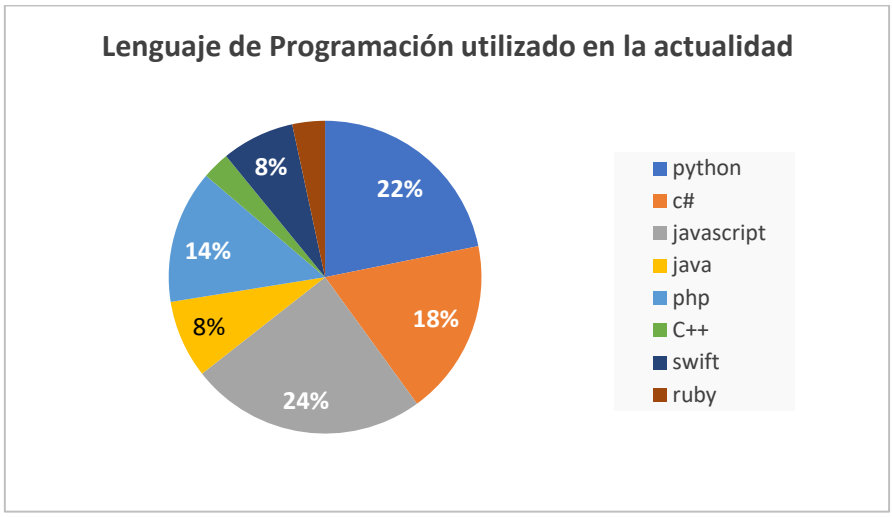
Figura 34. *Campos de Desarrollo*



12. ¿Qué lenguaje de programación estás utilizando actualmente?

Respecto a otros lenguajes de programación que están utilizando actualmente y según se muestra en la **Figura *35***, se obtuvo que: Python lo han utilizado 84 usuarios, lo que representa al 22%; C Sharp (C#) lo han utilizado 70 usuarios, lo que representa al 18%; JavaScript lo han utilizado 94 usuarios, lo que representa al 24%; Java lo han utilizado 31 usuarios, lo que representa al 8%; Php lo han utilizado 53 usuarios, lo que representa al 14% , C++ lo han utilizado 11 usuarios, lo que representa al 3%; Swift lo han utilizado 29 usuarios, lo que representa al 8%; Ruby lo han utilizado 13 usuarios lo que presenta al 3%. Con estos resultados se concluye que el lenguaje de programación que más se utiliza es Python.

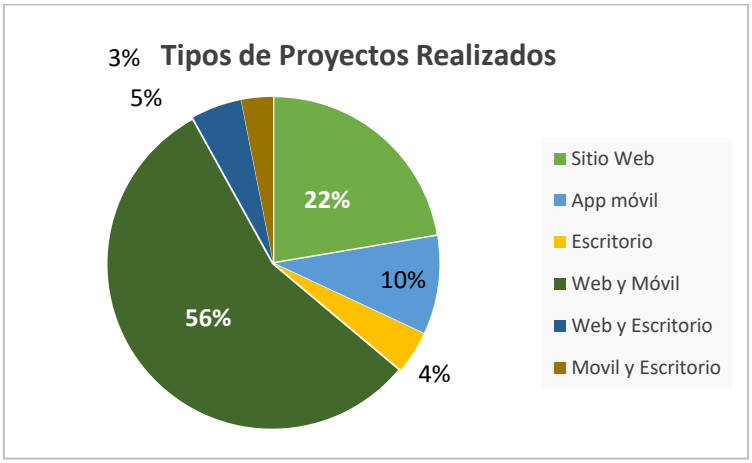
Figura 35. *Lenguaje de Programación utilizado*



13. ¿Qué tipo de proyecto te encuentras desarrollando actualmente?

Respecto al tipo de proyectos que se han desarrollado, según la **Figura *36***, muestra que: Sitio Web lo han seleccionado 86 usuarios, lo que representa al 22%; App móvil lo han seleccionado 37 usuarios, lo que representa al 10%; Escritorio lo han seleccionado 16 usuarios, lo que representa al 4%; Web y Móvil lo han seleccionado 215 usuarios, lo que representa al 56%; Web y Escritorio lo han seleccionado 19 usuarios, lo que representa al 5%; Móvil y Escritorio lo han seleccionado 12 usuarios, lo que representa al 3%. Con estos resultados se concluye que se realizan con mayor porcentaje los proyectos que se desarrollan para web y móvil.

Figura 36. *Tipos de Proyectos realizados*



Anexo *3.* Datos categóricos

ID	¿Cuál de los	¿Qué nivel d	¿De qué m	¿A qué si	En qu	¿En qué nivel consid	Cuando terminas tu	¿En que nivel consid	Para e	¿Qué rango de tiem	¿Has	¿Qué aspecto ha sid	De los siguientes ca	¿Qué tipo de aplicac
1	Flutter	Principiante	Equipo	Ambos	Alto	Muy Importante	ambiente preparado	Muy Importante	Medio	6 a 12 meses	Si	Visual(Interfaz animacic	Entretenimiento	App móvil
2	Flutter	Medio	Equipo	Ambos	Alto	Nada Importante	Configuración manual	Medianamente Importa	Medio	3 a 6 meses	Si	Visual(Interfaz animacic	Empresas	Escritorio
3	Xamarin	Alto	Individual	Android	Alto	Nada Importante	ambiente preparado	Medianamente Importa	Bajo	6 a 12 meses	Si	Visual(Interfaz animacic	Administración	Sitio Web
4	Ionic	Medio	Equipo	Android	Medio	Medianamente Importa	Configuración manual	Medianamente Importa	Bajo	6 a 12 meses	Si	Seguridad	Negocio	Web y Escritorio
5	Cordova	Medio	Equipo	Android	Bajo	Medianamente Importa	ambiente preparado	Medianamente Importa	Medio	12 meses o mas	Si	Seguridad	Entretenimiento	Web y Escritorio
6	Xamarin	Alto	Individual	Ambos	Bajo	Medianamente Importa	Configuración manual	Medianamente Importa	Medio	6 a 12 meses	Si	Visual(Interfaz animacic	Salud	Movil y Escritorio
7	ReactNative	Medio	Individual	Ambos	Medio	Nada Importante	ambiente preparado	Nada Importante	Bajo	6 a 12 meses	No	Duración de Batería	Administración	Escritorio
8	ReactNative	Alto	Equipo	iOS	Alto	Muy Importante	Configuración manual	Muy Importante	Bajo	6 a 12 meses	No	Rendimiento	Social	Web y Escritorio
9	Cordova	Principiante	Equipo	Ambos	Medio	Nada Importante	Configuración manual	Nada Importante	Bajo	3 meses o menos	No	Visual(Interfaz animacic	Negocio	Web y Móvil
10	Flutter	Alto	Individual	Ambos	Bajo	Muy Importante	Configuración manual	Muy Importante	Medio	6 a 12 meses	No	Visual(Interfaz animacic	Empresas	Sitio Web
11	Cordova	Medio	Individual	Android	Alto	Nada Importante	ambiente preparado	Nada Importante	Bajo	3 a 6 meses	No	Visual(Interfaz animacic	Administración	Movil y Escritorio
12	Ionic	Medio	Individual	Android	Alto	Nada Importante	ambiente preparado	Nada Importante	Bajo	3 meses o menos	Si	Visual(Interfaz animacic	Social	App móvil
13	Xamarin	Alto	Individual	Android	Medio	Nada Importante	ambiente preparado	Nada Importante	Medio	3 meses o menos	Si	Rendimiento	Administración	Web y Escritorio
14	ReactNative	Principiante	Equipo	Android	Medio	Medianamente Importa	Configuración manual	Nada Importante	Medio	6 a 12 meses	Si	Rendimiento	Administración	Escritorio
15	Xamarin	Alto	Individual	Android	Alto	Nada Importante	Configuración manual	Nada Importante	Bajo	12 meses o mas	Si	Duración de Batería	Negocio	Movil y Escritorio
16	Xamarin	Principiante	Individual	Android	Medio	Medianamente Importa	ambiente preparado	Nada Importante	Bajo	6 a 12 meses	Si	Seguridad	Empresas	Movil y Escritorio
17	Flutter	Alto	Individual	Android	Alto	Medianamente Importa	ambiente preparado	Medianamente Importa	Bajo	3 meses o menos	Si	Visual(Interfaz animacic	Educación	Movil y Escritorio
18	Ionic	Principiante	Individual	Android	Alto	Nada Importante	ambiente preparado	Nada Importante	Bajo	3 meses o menos	Si	Rendimiento	Negocio	Web y Escritorio
19	Xamarin	Principiante	Individual	Android	Medio	Nada Importante	ambiente preparado	Medianamente Importa	Medio	6 a 12 meses	No	Visual(Interfaz animacic	Educación	Web y Móvil
20	Cordova	Medio	Equipo	iOS	Medio	Medianamente Importa	Configuración manual	Medianamente Importa	Medio	6 a 12 meses	Si	Visual(Interfaz animacic	Entretenimiento	Sitio Web
21	Flutter	Medio	Individual	iOS	Medio	Muy Importante	Configuración manual	Muy Importante		3 meses o menos	Si	Rendimiento	Administración	Web y Escritorio
370	Xamarin	Medio	Equipo	Ambos	Medio	Medianamente Importa	ambiente preparado	Medianamente Importa	Alto	6 a 12 meses	Si	Visual(Interfaz animacic	Empresas	Web y Móvil
371	ReactNative	Principiante		Android	Bajo	Muy Importante	Configuración manual	Nada Importante	Medio	3 a 6 meses	Si	Rendimiento	Educación	Web y Móvil
372	Xamarin	Alto	Individual	Ambos	Alto	Muy Importante	ambiente preparado	Muy Importante	Alto	3 a 6 meses	Si	Visual(Interfaz animacic	Educación	Web y Móvil
373	ReactNative	Principiante	Equipo	Ambos	Medio	Medianamente Importa	ambiente preparado	Medianamente Importa	Medio	3 meses o menos	Si	Rendimiento	Empresas	Web y Móvil
374	ReactNative	Medio	Equipo	Ambos	Medio	Medianamente Importa	Configuración manual	Medianamente Importa	Medio	3 a 6 meses	No	Rendimiento	Entretenimiento	Web y Móvil
375	Cordova	Principiante	Equipo	Ambos	Bajo	Muy Importante	Configuración manual	Medianamente Importa	Medio	6 a 12 meses	Si	Visual(Interfaz animacic	Empresas	Web y Móvil
376	Ionic	Medio	Equipo	Ambos	Alto	Muy Importante	ambiente preparado	Muy Importante	Alto	3 a 6 meses	Si	Rendimiento	Empresas	Web y Móvil
377	Ionic	Principiante	Equipo	Ambos	Bajo	Nada Importante	Configuración manual	Muy Importante	Medio	3 a 6 meses	No	Duración de Batería	Empresas	App móvil
378	Ionic	Principiante	Individual	Android	Bajo	Muy Importante	Configuración manual		Bajo	3 a 6 meses	No	Seguridad	Social	Web y Móvil
379	Ionic	Principiante	Equipo	iOS	Medio	Muy Importante	Configuración manual	Medianamente Importa	Medio	6 a 12 meses	No	Duración de Batería	Salud	Web y Móvil
380	Ionic	Medio	Equipo	Android	Medio	Medianamente Importa	Configuración manual	Medianamente Importa	Medio	3 a 6 meses	No	Duración de Batería	Educación	Web y Móvil
381	Cordova	Principiante	Individual	Ambos	Bajo	Muy Importante	Configuración manual	Muy Importante	Alto	3 a 6 meses	Si	Rendimiento	Empresas	Web y Móvil
382	Xamarin	Medio	Equipo	Ambos	Bajo	Muy Importante	Configuración manual	Medianamente Importa	Alto	12 meses o mas	No	Duración de Batería	Empresas	Web y Móvil
383	Ionic	Medio	Individual	Ambos	Medio	Muy Importante	Configuración manual	Muy Importante	Alto	6 a 12 meses	Si	Rendimiento	Empresas	Web y Móvil
384	Flutter	Medio	Equipo	Ambos	Medio	Nada Importante	ambiente preparado	Nada Importante	Bajo	3 a 6 meses	Si	Duración de Batería	Negocio	Escritorio
385	Ionic	Alto	Equipo	Ambos	Alto	Muy Importante	ambiente preparado	Muy Importante	Alto		Si		Administración	Web y Móvil

Fase de Preparación de la Data

Anexo *4.* Datos categóricos ordinales

```
In [4]: print(df)
     Experiencia  Modo  SO  Documentacion   ...  Campos  Lenguaje  Tipo  Entornos
0              1     1   3              4  ...       1         2     1         1
1              3     2   3              1  ...       2         1     1         1
2              4     1   1              3  ...       3         3     1         4
3              3     2   1              2  ...       4         7     1         3
4              3     2   1              4  ...       1         7     1         5
..           ...   ...  ..            ...  ...     ...       ...   ...       ...
380            1     1   3              3  ...       2         3     6         5
381            3     2   3              3  ...       2         3     6         4
382            3     1   3              4  ...       2         1     6         3
383            3     2   3              1  ...       4         2     3         1
384            3     2   3              3  ...       3         4     6         3

[385 rows x 14 columns]
```

Librerías

```
#Librerías
#Carga del dataset
archivo="/home/adriana/Desktop/IA/Dataset9.csv"
df=pd.read_csv(archivo)
#importar la librería de clasificación de vecinos cercanos knn
from sklearn.neighbors import KNeighborsClassifier
#impotar la librería de clasificación con red neuronal
from sklearn.neural_network import MLPClassifier
from sklearn import tree
#importar la librería para grabar la máquina
import joblib
#importar las librerías para realiz
import numpy as np
import pandas as pd
```

Fase de Modelamiento

```
#Separar la data en data de entrenamiento y data de prueb
x_train,x_test,y_train,y_test=train_test_split(X,y)

## para set de entrenamiento y prueba
x_train.shape ## elementos de entrenamiento

#Correlación de los datos
df.corr()
```

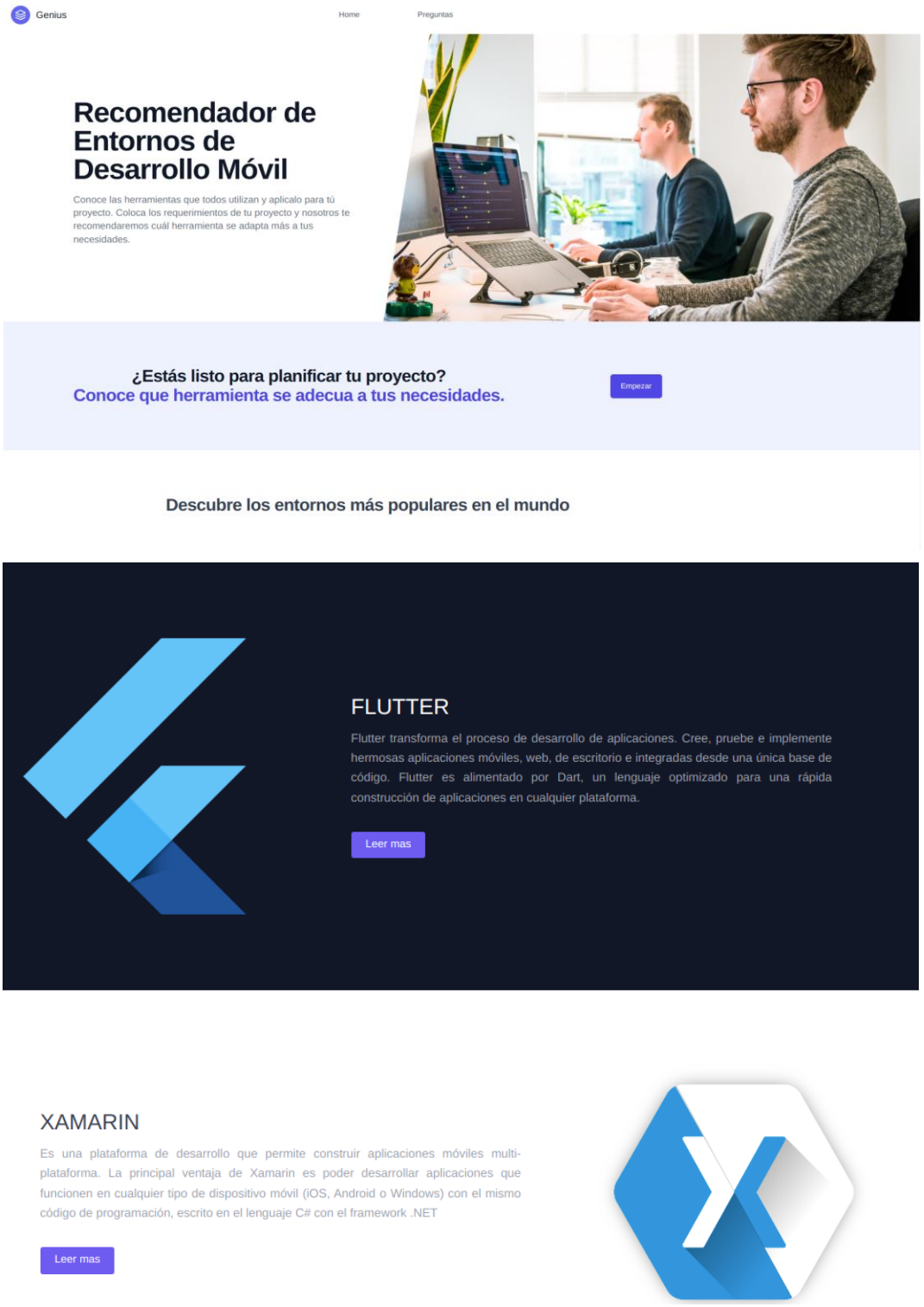
```
#Crear la máquina de aprendizaje supervisado
df=DecisionTreeClassifier(max_depth=8)
knn=KNeighborsClassifier(n_neighbors=9)
neural_net=MLPClassifier(max_iter=1000,hidden_layer_sizes=800)

#Entrenar la máquina
df.fit(x_train,y_train)
knn.fit(x_train,y_train)
neural_net.fit(x_train,y_train)

#Crear la máquina de aprendizaje supervisado
df=DecisionTreeClassifier(max_depth=9)
knn=KNeighborsClassifier(n_neighbors=9)
neural_net=MLPClassifier(max_iter=30000,hidden_layer_sizes=10000)
#Entrenar la máquina
df.fit(x_train,y_train)
knn.fit(x_train,y_train)
neural_net.fit(x_train,y_train)

#Crear la máquina de aprendizaje supervisado
df=DecisionTreeClassifier(max_depth=10)
knn=KNeighborsClassifier(n_neighbors=9)
neural_net=MLPClassifier(max_iter=9000,hidden_layer_sizes=20000)
#Entrenar la máquina
df.fit(x_train,y_train)
knn.fit(x_train,y_train)
neural_net.fit(x_train,y_train)
```

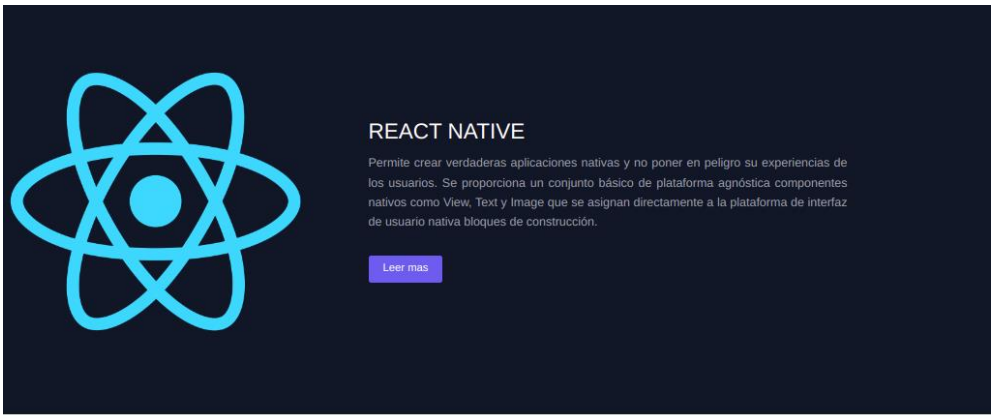
Fase de Evaluación

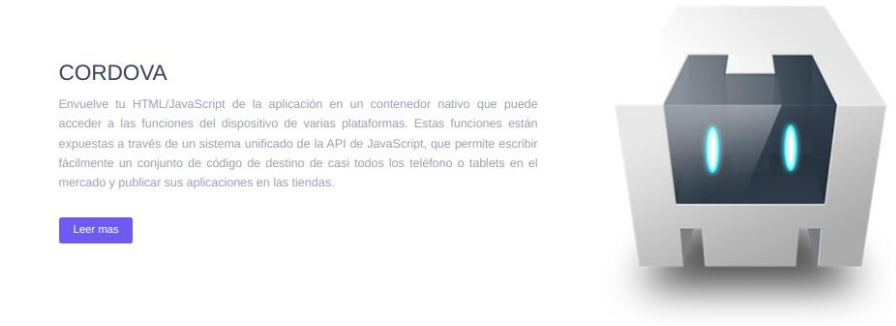
```
#Probar la eficiencia de la máquina
print(df.score(x_test,y_test))
print(knn.score(x_test,y_test))
print(neural_net.score(x_test,y_test))

#Resultados
print(df.score(x_train,y_train))
print(knn.score(x_train,y_train))
print(neural_net.score(x_train,y_train))
```

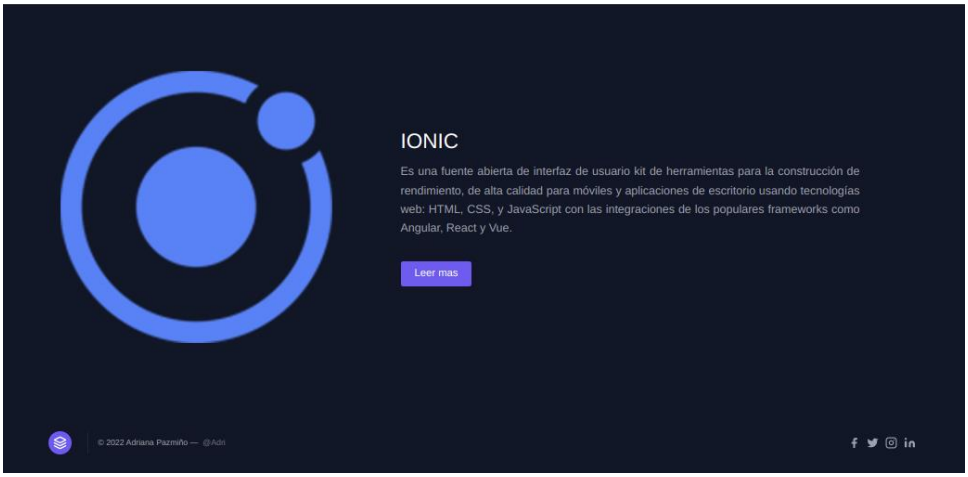
Fase de Despliegue

Anexo 5. Diseño de Sitio web

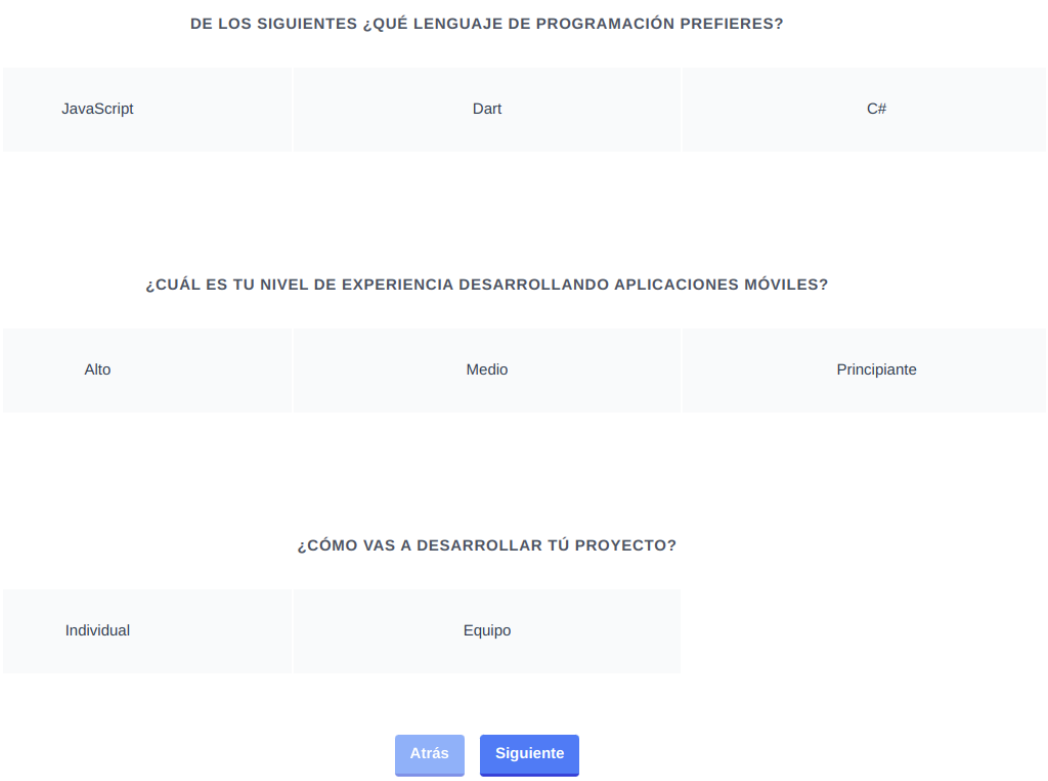



REACT NATIVE
Permite crear verdaderas aplicaciones nativas y no poner en peligro su experiencias de los usuarios. Se proporciona un conjunto básico de plataforma agnóstica componentes nativos como View, Text y Image que se asignan directamente a la plataforma de interfaz de usuario nativa bloques de construcción.
Leer mas

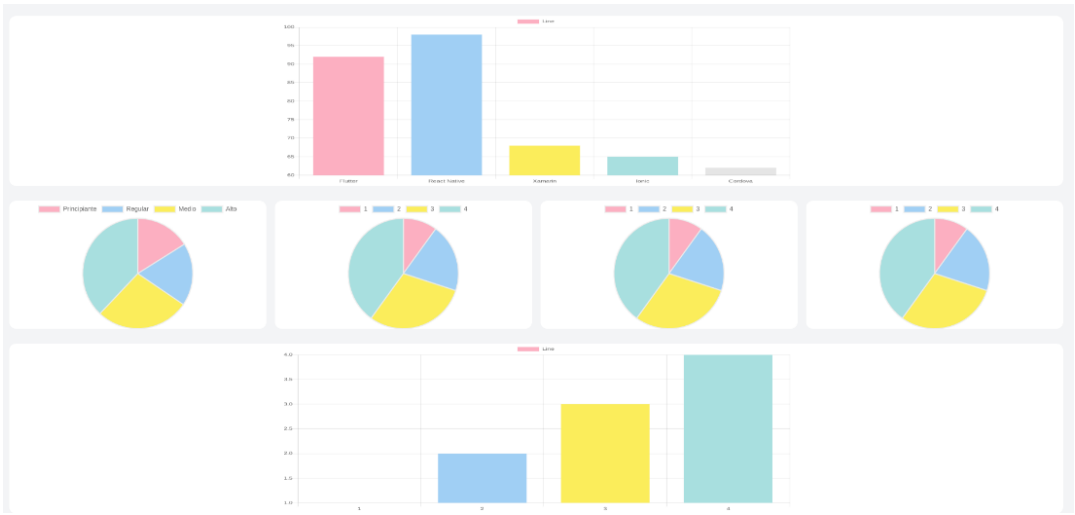

CORDOVA
Envuelve tu HTML/JavaScript de la aplicación en un contenedor nativo que puede acceder a las funciones del dispositivo de varias plataformas. Estas funciones están expuestas a través de un sistema unificado de la API de JavaScript, que permite escribir fácilmente un conjunto de código de destino de casi todos los teléfono o tablets en el mercado y publicar sus aplicaciones en las tiendas.
Leer mas


IONIC
Es una fuente abierta de interfaz de usuario kit de herramientas para la construcción de rendimiento, de alta calidad para móviles y aplicaciones de escritorio usando tecnologías web: HTML, CSS, y JavaScript con las integraciones de los populares frameworks como Angular, React y Vue.
Leer mas
© 2022 Adriana Pazmiño — @Adri

Sección de preguntas



Sección de estadísticas



Información de Entornos