SOLUCIÓN TECNOLÓGICA

ALTERNATIVA RECOMENDABLE

.NET No

Java No

VENTAJAS

\*Framework que permite desarrollo web, app de escritorio y app movil para iOS, Android y Windows Phone.

\*Desarrolla aplicaciones independiente de la arquitectura física (procesador) y del sistema operativo.

\*Multipropósito y multiplataforma.

\*Funciona con protocolo XML similar en sus sintaxis a HTML.

\*Framework que permite montar sobre el sistema operativo para ejecutar aplicaciones (diferentes lenguajes) de forma segura.

\*Actualmente pertenece a Oracle y tiene más de 3.000.000.000 de dispositivos y 12.000.000.000 de desarrolladores.

DESVENTAJAS

\*Es utilizado por grandes empresas, por lo cual, tiene un valor elevado y es difícil que sea recomendable para un startup.

\*Es utilizado por grandes empresas, por lo cual, tiene un valor elevado y es difícil que sea recomendable para un startup.

ELECCIÓN

Swift

No \*Lenguaje multipropósito y multiparadigma.

\*Sintaxis simple.

\*Enfocado a objetos, protocolos, funcional y programacion imperativa.

\*Las versiones de las app para los dispositivos se ajustan facilmente con Xcode.

\*A pesar de ser poco de la participación de mercado, los usuarios de iOS son los que más compran apps, y el 90% utiliza las últimas versiones de las mismas.

\*Se integra con framework Cocoa y Cocoa Touch.

\*Enfocado a aplicaciones del ecosistema Apple.

\*Mayoría de usuarios a nivel mundial son Android.

Ruby

Si \*Lenguaje dinámico y de código abierto que se enfoca en la simplicidad y productividad.

\*Todos los elementos son considerados objetos (asignables propiedades y acciones).

\*Muy flexible y portable.

\*Lenguaje poco conocido y utilizado, por lo cual Si cuesta encontrar desarrolladores.

Ruby on Rails Si

Python No

\*Framework basado en Ruby para el desarrollo de aplicaciones web (código abierto).

\*Sigue la filosofía DRY (No te repitas).

\*Permite prototipar ideas rapidamente hasta desarrollar apps complejas.

\*Ideal para gente que nunca ha programado.

\*Tiene una sólida comunidad de desarrolladores, liderados por la PSF (Python Software Foundation)

=> soporte.

\*Principalmente usado en la comunidad científica por su capacidad de procesar procedimientos matemáticos complejos.

\*No tiene una app centralizada de manejo de Si contenidos como Django.

\*Consume mucho recursos por la complejidad de sus operaciones.

\*Curva de aprendizaje es muy grande.

\*No todos los hosting pueden hospedar apps en Python.

Django

Ionic

No \*Framework para aplicaciones desarrolladas con Python.

\*Usa patrón de diseño Modelo Vista Controlador (MVC).

\*Busca facilitar la creación de sitios web dinámicos y complejos basados en re-uso, conectividad desarrollo rápido y DRY.

\*Tiene una app incorporada de manejo de contenido, que puede administrar varias páginas a partir de una instalación.

No \*Framework gratuito de reciente creación para desarrollo de apps basadas en HTML5, CSS y Javascript.

\*Es de código abierto.

\*Trabaja con AngularJS (arquitectura robusta para apps).

\*Consume muchos recursos por usar Python.

\*Usa lenguaje híbrido, por lo cual no es del mismo nivel que una app nativa.

\*No recomendable para proyectos de gran envergadura.

\*Limitaciones con iOS:

Xamarin

No \*Lenguaje para desarrollar apps móviles multiplataforma y se compila de forma nativa (alto rendimiento).

\*Tiene elementos de seguridad que evitan comportamientos erráticos del código.

\*Acceso limitado a bibliotecas.

\*Alto costo.

\*Pocos desarrolladores.

Kotlin

No \*Lenguaje para Android dirigido a desarrolladores backend.

\*100% compatible con librería Java.

\*Funcional con codificación simplificada.

\*Dificil comprensión inicial del código.

\*Poco soporte y tiempos de ejecución largos.