

Universidad Tecnológica de El Salvador



FACULTAD:	Facultad de Informatica y Ciencias Aplicadas
DOCENTE	Ing. José Guillermo Rivera
MATERIA	Técnicas de calidad de Software
TEMA	Desarrollo de Aplicaciones Moviles con FrameWork Flutter

GRUPO 4

CARNET	NOMBRE
17-1673-2000	José Alfredo Hernández Ortiz
17-3514-2018	Karla Alexandra Reyes Alfaro
25-4970-2016	Vladimir Benjamín Hernández Pérez
25-0686-2015	Fátima Michelle Molina García
25-5902-2015	Víctor Manuel Rodríguez Aguiluz

ÍNDICE

CONTENIDO	PAGINA
Introducción	4
CAPITULO I: Especificación del proyecto	5
a. Situación actual	6
i. Antecedentes	
ii. Descripción del problema	
iii. Planteamiento del problema	
iv. Hipotesis	
b. Tema	9
c. Sistema de objetivos	10
i. Objetivo general	
ii. Objetivos espedificos	
d. Alcances	11
i. Alcance 1	
1. Producto 1	
ii. Alcance 2	
1. Producto 2	
iii. Alcance 3	
1. Producto 3	
Limitantes	
e. Justificación	12
f. Cronograma	13
g. Presupuesto	14
CAPITULO II: Especificación del proyecto	15
a. Metodología de trabajo	16
Taller de requerimientos	
Gestion de usuarios	
Visualizacion de datos	
Informes (No existen)	
Metodologia a utilizar para desarrollar el prototipo	
b. Descripción de la propuesta de solución	22
c. Descripción de la tecnología disponible	23
d. Evaluación de tecnología disponible	25
e. Diagrama arquitéctonico de la solución	29
f. descripcion de cada componente de la solucion	30

CAPITULO III: Estrategia de implementación del proyecto	37
a. Estrategia de implementación	38
b. Presupuesto de implementación	42
c . Análisis de resultados	43
Conclusiones	45
Recomendaciones	46
Bibliografía	47
Anexos	48
Anexo1 - Main	49
Anexo2 - Lista	52
Anexo3 - Login	54
Anexo4 - Modificar Precios	56
Anexo5 - User	58
Anexo6- UsersClass	64

INTRODUCCION

El presente proyecto, pretende dar solución a un problema inherente a todo conductor de vehículos automotores, sean automóviles, camiones o motocicletas, o bien cualquier otro vehículo que dependa de gasolina o diésel.

Este factor común, es la economía.

Todo conductor, busca los mejores precios en la zona donde se encuentre, a la hora de cargar combustible.

Una solución viable y económica, es el desarrollo de una aplicación que facilite al usuario final, una lista de gasolineras dentro de la zona, con sus precios al día.

Tal situación otorga al usuario la ventaja de poder elegir el mejor precio, puesto que es conocimiento de todos, que el precio de los combustibles varía de una gasolinera a otra, por cercanas que estas se encuentren.

Una diferencia de centavos puede parecer risible, pero al contabilizar un año, puede notarse un ahorro significativo.

CAPITULO I

Especificación del proyecto

SITUACIÓN ACTUAL

Todo conductor conoce, que los precios de los combustibles fluctúan constantemente, debido a una serie de factores especialmente en el mercado internacional.

Tal fluctuación afecta los bolsillo de forma directa.

Cuál es el problema más grande es que no se tiene conocimiento en tiempo real sobre esta fluctuación de precios sino hasta el momento de visitar una gasolinera.

Generalmente el gobierno anuncia incremento o baja de precios pero este incremento o baja de precios no es estándar para todas las gasolineras ya que cada una maneja un rango de precios muy diferente que normalmente puede variar en apenas un par de centavos.

Sin embargo es de notar que esa diferencia de 2 o 3 centavos es un costo acumulable que al final de 1 año puede ser un valor bastante considerable o en su defecto un ahorro bastante considerable.

La única forma de contar con ese ahorro es conociendo en tiempo real la diferencia de precios entre las gasolineras de la zona.

A este día no existe una herramienta que pueda mostrar al consumidor los precios de los combustibles en el área de acción en tiempo real en consecuencia tiene que arriesgarse y visitar la gasolinera que él por tradición conoce que maneja precios un poco más cómodos.

Situación que no siempre es certera es más en la mayoría de ocasiones no lo es.

La situación sería un tanto diferente si se contara con una herramienta que muestra en tiempo real el rango de precios de las gasolineras de la zona para ayudar al consumidor y ahorrar en el consumo de combustibles.

ANTECEDENTES

Durante los últimos años, los precios de los combustibles han variado constantemente, suben y baja como las olas del mar.

Para todo conductor es de suma importancia mantenerse en conocimiento de estas variaciones, puesto afecta su economía.

Muchas personas utilizan siempre la misma gasolinera, debido a ciertas preferencias que estos tienen.

También los hay aquellos siempre están a la búsqueda de los precios más competitivos.

Existe otro grupo que, por la naturaleza de sus labores, necesitan desplazarse por diferentes zonas del país, para estos muy difícil, sino imposible mantenerse al tanto de los precios.

Como ya se mencionó, es un problema que afecta directamente su economía.

DESCRIPCIÓN DEL PROBLEMA

El asunto estriba en la necesidad de conocer los precios de combustibles en la zona de desplazamiento.

Evidentemente que gasolinera dispone de los mejores precios a la hora de cargar combustible.

Hasta el momento no existe una herramienta que disponga de esta información.

Especialmente al desplazarse fuera del área habitual, los conductores no disponen de información que le representa un ahorro o en su defecto un incremento en sus gastos de combustible.

Considerando la tecnología actual y el hecho que un muy pequeño porcentaje de usuarios no utilizan teléfonos inteligentes, es de notar la inexistencia de una aplicación con este fin.

PLANTEAMIENTO DEL PROBLEMA

Se espera desarrollar una aplicación que aporte la información sobre los precios de combustibles.

Se utilizará tecnología para el SO Android, y lenguaje Java.

La aplicación dispondrá de dos marcos de trabajo:

ESPACIO PARA PROPIETARIOS DE GASOLINERAS:

Estará disponible únicamente para administradores de las gasolineras que se asocien, desde este marco, podrán administrar la información correspondiente en el momento que esta deba modificarse.

Deberá autenticarse para ingresar.

ESPACIO PARA USUARIO FINAL:

Espacio para usuario final, no necesita autenticación.

Solo permite consultar información acerca de las gasolineras en la zona y rango de precios.

El usuario final no puede realizar modificaciones.

HIPÓTESIS

Se desarrollará una aplicación que muestre en tiempo real los rangos de precios de las gasolineras del área metropolitana de San Salvador.

Esta aplicación resolverá el problema que enfrentan todos los consumidores de combustibles al no conocer los precios en las diferentes estaciones de servicio y en consecuencia incurrir en la visita de una estación que podría disponer de los precios más altos al no conocer que otras estaciones manejan precios más cómodos.

Se espera que las estaciones de combustible al publicar los precios de los mismos puedan atraer más clientes.

Se espera que el cliente final pueda conocer los precios de los combustibles que consume a diario publicados por las diferentes estaciones y se les facilite elegir la mejor opción

TEMA

Se propone el presente proyecto.

APLICACION PARA LA GESTION DE LOS PRECIOS DE COMBUSTIBLE DE LA ZONA
METROPOLITANA DE SAN SALVADOR.

PETROLIN
MONITOREO DE PRECIOS DE COMBUSTIBLES

OBJETIVOS

OBJETIVO GENERAL

Desarrollar una aplicación móvil que administre el flujo de precios de combustibles.

OBJETIVOS ESPECIFICOS

Desarrollar una aplicación que permita a los propietarios de las gasolineras del área metropolitana de San Salvador, mostrar al público los precios de combustibles, actualizados al día.

Implementar una aplicación que permita a los usuarios de las gasolineras del área metropolitana de San Salvador, consultar los precios de los combustibles, actualizados al día.

ALCANCES

ALCANCE 1

ADMINISTRADORES DE GASOLINERAS.

Estos tendrán las credenciales correspondientes para administrar la información concerniente a la gasolinera que administran.

ALCANCE 2

USUARIO FINAL.

El usuario final podrá consultar información filtrándola por departamento -> municipio, en orden de menor a mayor rango de precios.

ALCANCE 3

OTROS NEGOCIOS

Esta aplicación será utilizada por un número considerable de usuarios, la cual dispone de una interfaz sobria y minimalista, por tanto, dispone de un espacio suficiente para colocar publicidad.

Este rubro es de tomarlo en consideración, no necesariamente debe aplicarse desde sus primeras versiones, sin embargo está contemplado para futura implementación.

LIMITANTES

En todo proyecto a desarrollar existen limitaciones a considerar, para el presente caso se puede considerar que, es imperante el uso de las tecnologías para dispositivos móviles, las condiciones propias del equipo de trabajo y tecnología disponible.

Existen muchos lenguajes que permiten el desarrollo de aplicaciones móviles, para el presente caso, se utilizará el lenguaje Java, con Android Studio.

Se puede anumerar al momento las siguientes limitantes del equipo de trabajo.

- Conocimiento de Sistema Operativo Android
- Conocimiento del lenguaje Java
- Capacidad procesamiento de los equipos disponibles.
- Conocimiento de la tecnología en cuestión por parte del equipo de trabajo.

JUSTIFICACIÓN

Existe oportunidad de negocio, que se puede abordar de dos áreas muy diferentes a notar.

Gasolineras

Uso por suscripción. Es posible cobrar una cuota por suscripción a los propietarios de las gasolineras, puesto que estos los directamente beneficiados, aunque existe el inconveniente que estos tienen políticas establecidas para la gestión de sus precios, lo que los deja en franca desventaja frente a aquellos que disponen de mejores ofertas.

Publicidad

Es factible ofrecer publicidad a terceros, disponiéndola de tal forma que no afecte al usuario final, sin embargo, que sea visible para este.

Este rubro es más probable.

CRONOGRAMA

	AGO	SEP	OCT	NOV	DIC
Documentación (proyecto)	X				
Propuesta de diseño	X				
Selección de Sistema Operativo	X				
Selección de FrameWork	X				
Versión Alfa		X			
Versión Beta			X		
Versión Final				X	
Testing		X	X	X	
Presentación del proyecto				X	

PRESUPUESTO (CORREGIDO Y ACTUALIZADO)

EQUIPO DE TRABAJO	CANTIDAD	\$ POR HORA	HORAS / D	HORAS / MES	TOTAL PROYECTO	TOTAL \$
Coordinador	1	\$ 30.00	2	8	32	\$ 960.00
Diseñador de bases de datos	1	\$ 25.00	2	8	32	\$ 800.00
Diseñador UI	1	\$ 25.00	2	8	32	\$ 800.00
Programadores	1	\$ 25.00	2	8	32	\$ 800.00
TOTAL						\$ 3,360.00

EQUIPO TECNOLÓGICO	CANTIDAD	COSTO POR USO	TOTAL
Computadoras	5	\$ 5.00	\$ 25.00
Local de trabajo	5	\$ 50.00	\$ 250.00
Electricidad	1	\$ 100.00	\$ 100.00
Agua	1	\$ 10.00	\$ 10.00
TOTAL			\$ 385.00

COSTO TOTAL ESTIMADO	
TOTAL	\$ 3,745.00
IMPREVISTOS	\$ 1,123.50
TOTAL	\$ 4,868.50

CAPITULO II

Analisis y diseño de la propuesta de solución

METODOLOGÍA DE TRABAJO

TALLER DE REQUERIMIENTOS

1. Necesidades de negocio o sistema.

Todo consumidor necesita conocer los precios de los productos que consume y los combustibles para automotores, no son la excepción.

Es de conocimiento público, que los precios de los combustibles fluctúan constantemente.

Esta condición vuelve imperante conocer la fluctuación de precios en tiempo real en las diferentes estaciones de servicio del área metropolitana de San Salvador.

Se requiere:

- Conocer los precios de combustibles en tiempo real.
- Conocer la ubicación geográfica de las estaciones de servicio.

2. Reglas de negocio.

Administrador:

Los administradores de las estaciones de servicio deberán crear un usuario para gestionar los precios de los combustibles.

Necesitará un código de verificación para crear el usuario.

La aplicación no necesita un CRUD completo, al crearse el usuario, se crearan la tabla de precios, un registro con los datos necesarios, este registro es único para cada estación y solamente podrá actualizar la información No se requiere histórico de precios.

3. Visión del producto.

Se espero una aplicación sencilla, fácil de usar.

Pantalla de bienvenida.

Muestra al usuario un mensaje de bienvenida.

Dispone de dos opciones, la una para listar precios actuales y la otra para gestionar los precios.

La opción de gestionar precios requiere inicio de sesión.

Si el usuario no está registrado, dispone de opción para crear las credenciales.

4. Requerimientos (usuario, sistema, hardware, etc.)

Usuario final:	No requiere autenticación.
Administrador:	Requiere autenticación.
Sistema Operativo:	Android 10 - API 34
Plataforma de desarrollo:	Android Studio

Lenguaje:	Java
Dispositivos compatibles:	Smartphone Android 10

GESTIÓN DE USUARIOS

USUARIO FINAL

El usuario final, no requiere de un perfil.

El usuario final realiza consultas anónimas, considerando que su único interés, es conocer los precios de combustibles, en relación a los precios más cómodos para su economía.

Administradores de estaciones.

Los administradores, si requieren de un perfil, puesto que estos son los responsables de actualizar los precios de los combustibles, cuando así sea requerido.

Para actualizar los precios, es requerido el inicio de sesión.

Para crear el usuario, es necesario un código de activación, con el propósito de evitar que terceros maliciosos creen usuarios y registren datos falsos

VISUALIZACIÓN DE DATOS

Al iniciar la aplicación se muestra listado de las 10 estaciones que ofrecen los mejores precios, en orden ascendente. (Nombres de estaciones).

Al pulsar sobre el nombre de la estación, se despliega nueva pantalla que muestra los datos generales de esta y los precios de los combustibles que sirve, a saber:

Gasolina Súper, Gasolina Regular, Diésel.

INFORMES

La presenta aplicación, no dispone de informes.

METODOLOGÍA A UTILIZAR PARA DESARROLLAR EL PROTOTIPO

DETECCIÓN DE VARIABLES:

Ya se ha mencionado, que no existe una aplicación que reporte en tiempo real los precios de combustibles.

Principalmente debido a que las estaciones consideran poseer una cartera de clientes fieles. Situación que no es muy real, pues si bien es cierto que los conductores prefieren una marca, también están a la expectativa de los mejores precios.

GENERACIÓN DE CONCEPTOS:

Se espera el desarrollo de una aplicación que muestre a los consumidores el listado de las gasolineras que disponen de los mejores precios en orden ascendente.

Al pulsar sobre la gasolinera elegida, se mostrará los detalles de la misma y los precios de los combustibles.

DEFINICIÓN DE LA PROPUESTA:

La aplicación se desarrollará en el lenguaje Java.

Las consultas realizadas por los usuarios será anónimas, no necesitan registrarse.

Los administradores de estaciones de servicio, deberán crear un usuario para registro y posterior actualización de los precios.

La aplicación no generará reportes.

DISEÑO DE PROTOTIPOS:

Administradores de estaciones.

Los administradores, si requieren de un perfil, puesto que estos son los responsables de actualizar los precios de los combustibles, cuando así sea requerido.

Para actualizar los precios, es requerido el inicio de sesión.

Para crear el usuario, es necesario un código de activación, con el propósito de evitar que terceros maliciosos creen usuarios y registren datos falsos

Este prototipo muestra una pantalla con dos secciones principales. La primera sección, titulada 'USUARIO', contiene un campo de texto para el nombre de usuario y un botón 'Iniciar Sesión'. La segunda sección, titulada 'CONTRASEÑA', contiene un campo de texto para la contraseña y un botón 'Crear Usuario'. Ambas secciones tienen un fondo azul claro y los campos de texto son blancos.

Este prototipo muestra una pantalla titulada 'Crear usuario'. Contiene campos de texto para 'Empresa', 'Direccion', 'Usuario', 'Pass' y 'Verificar'. Debajo de estos campos hay un botón 'Registrar'.

Visualización de datos

Al iniciar la aplicación se muestra listado de las 10 estaciones que ofrecen los mejores precios, en orden ascendente. (Nombres de estaciones).

Al pulsar sobre el nombre de la estación, se despliega nueva pantalla que muestra los datos generales de esta y los precios de los combustibles que sirve, a saber:

Gasolina Súper, Gasolina Regular, Diésel.

Este prototipo muestra una pantalla titulada 'Lista en orden descendente por precios'. Contiene una lista de estaciones: Estación A, Estación B, Estación C, Estación D, Estación E, Estación F y Estación G. Cada estación tiene un botón a su lado.

Este prototipo muestra una pantalla titulada 'GASOLINERA X'. Contiene una sección 'UBICACION' y una tabla de precios. La tabla tiene dos columnas: 'Producto' y 'Precio'. Se muestran tres filas de datos.

Producto	Precio
Producto	Precio
Producto	Precio
Producto	Precio

Informes

La aplicación, no dispone de informes.

METODOLOGÍA A UTILIZAR PARA DESARROLLAR EL PROTOTIPO

DETECCIÓN DE VARIABLES:

Ya se ha mencionado, que no existe una aplicación que reporte en tiempo real los precios de combustibles.

Principalmente debido a que las estaciones consideran poseer una cartera de clientes fieles. Situación que no es muy real, pues si bien es cierto que los conductores prefieren una marca, también están a la expectativa de los mejores precios.

GENERACIÓN DE CONCEPTOS:

Se espera el desarrollo de una aplicación que muestre a los consumidores el listado de las gasolineras que disponen de los mejores precios en orden ascendente.

Al pulsar sobre la gasolinera elegida, se mostrará los detalles de la misma y los precios de los combustibles.

DEFINICIÓN DE LA PROPUESTA:

La aplicación se desarrollará en el Android Studio, con el lenguaje Java.

Las consultas realizadas por los usuarios será anónimas, no necesitan registrarse.

Los administradores de estaciones de servicio, deberán crear un usuario para registro y posterior actualización de los precios.

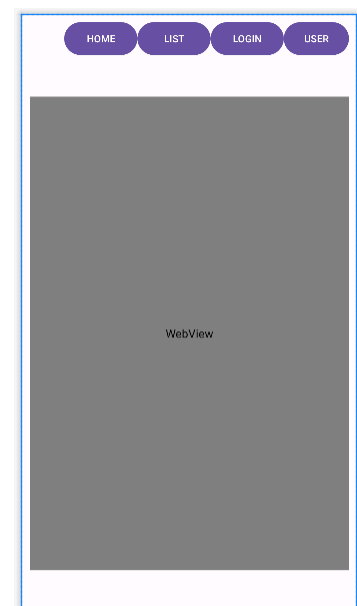
La aplicación no generará reportes.

DISEÑO DE PROTOTIPOS:

PANTALLA INICIAL. (HOME)

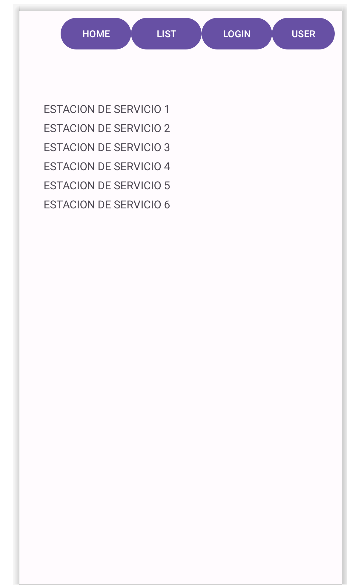
Dispone de los botones principales de navegación.

Muestra un sitio web, que podría contener publicidad.



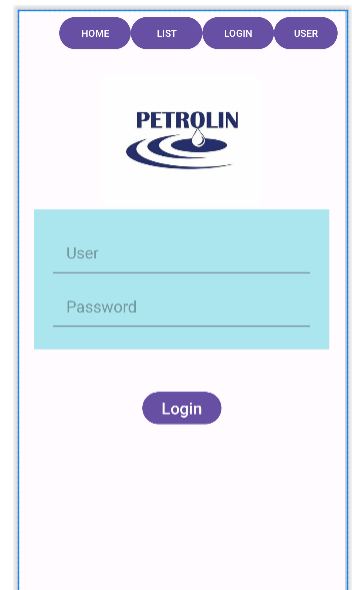
LISTADO DE ESTACIONES DE SERVICIO. (LIST)

Despliega el listado de las estaciones de servicio que disponen los mejores precios, en orden ascendente.



INICIO DE SESIÓN. (LOGIN)

Inicio de sesión para usuarios registrados. (Estaciones de servicio)



CREAR USUARIO. (USER)

Creación de nuevos usuarios.

Se requiere un código de validación.

HOME LIST LOGIN USER

CREAR USUARIOS

PETROLIN

Empresa

Direccion

Usuario

Contraseña

Codigo verificacion

CONFIRMACIÓN DE CREACIÓN USUARIO

Confirmación de creación de nuevo usuario



Home

Update Cancel

Gasolina Super: 0.00

Gasolina Regular: 0.00

Diesel: 0.00

ACTUALIZACION DE PRECIOS. (UPDATE)

Actualizacion de los precios de combustibles.

DESCRIPCION DE LA PROPUESTA.

DISEÑADO PARA:

Empresas Distribuidoras de Combustibles Para Automotes y Equipos Similares:

Gasolinera Puma.

Gasolinera Texaco.

Gasolinera Uno.

EQUIPO DE DESARROLLO:

La Aplicación Petrol In, ha sido diseñada considerando la necesidad de los conductores de automotores, del área metropolitana de San Salvador, por conocer los mejores precios de los combustibles que consumen a diario.

SERVICIOS:

La Aplicación cuenta con dos servicios únicamente.

Usuario Final.

Servicio que muestra al consumidor, los precios de las estaciones de servicio, así como su ubicación.

Administrador.

Servicio para los administradores de las estaciones de servicio, que permite actualizar los precios de combustibles para ser presentados a los consumidores.

Inicio de Sesión.

Es necesario que los administradores inicien sesión para actualizar los precios de combustibles.

Creación de Usuarios.

Los Administradores podrán crear sus usuario, con el nombre que mejor les apetezca, considerando que este debe ser único; Para la creación de usuarios, se requiere de un código de activación, proporcionado por el Administrador de la aplicación.

COSTOS:

El uso de la aplicación, no representa un costo para las estaciones de servicio, sin embargo es de considerar que la aplicación puede presentar publicidad de terceros.

DESCRIPCIÓN DE LAS TECNOLOGÍAS DISPONIBLES

LENGUAJES DE PROGRAMACIÓN.

Los lenguajes para desarrollo de aplicaciones móviles, han evolucionado mucho.

Ahora disponemos de lenguajes más eficientes y mayores capacidades para gestionar diferentes servicios.

LENGUAJE DART

Es pues el ejemplo del lenguaje Dart.

Este lenguaje no es nuevo, tiene ya algunos años en servicio, creado por Google, pero recientemente se ha apreciado sus capacidades, tiene la cualidad de ser un lenguaje multiplataforma, es decir funciona igual en Windows, Mac, Linux ó Android.

Consume muy pocos recursos.

También sigue en el mercado el tan conocido lenguaje Java.

Este lenguaje es muy eficiente, también es multiplataforma, tiene la ventaja de una baja curva de aprendizaje.

LENGUAJE JAVA

El lenguaje Java, ya tiene un tiempo en el mercado, es multiplataforma, de uso libre, tiene la capacidad para desarrollar aplicaciones de escritorio, para la web y evidentemente para dispositivos móviles.

FRAMEWORK

Flutter, es un framework desarrollado por Google, para el desarrollo de aplicaciones móviles multiplataforma.

Cuenta con tecnología, que tampoco es joven pero hoy en día suena como si lo fuera.

Los componentes que utiliza para dar forma a las pantallas, se denominan widgets.

En flutter todo es un widget. Que implica esto, pues estriba en su modularidad.

Cada widget es independiente del otro, pero interactúan entre sí, creando aplicaciones atractivas y con mayor interactividad.

ANDROID STUDIO

Android Studio es un IDE, ya conocido, también desarrollado por Google, con grandes bondades para el desarrollo de aplicaciones móviles.

VS CODE

Visual Studio Code, es un IDE desarrollado por Microsoft.

Es ideal para el desarrollo de aplicaciones en una diversidad de lenguajes. Especialmente para aquellos en los cuales predomina el desarrollo por consola.

EQUIPO INFORMÁTICO.

Computadora portátil:	Asus Sephiros.
Sistema Operativo:	Windows 11
Procesador:	Ryzen 7
GPU:	NVidia GETFORCE GTX
RAM:	16 Gb
IDE para desarrollo:	Android Studio Giraffe, VS Code

EVALUACIÓN DE LAS TECNOLOGÍA DISPONIBLE

LENGUAJE DE PROGRAMACION JAVA

Características de Java

- Simplicidad.
- Portabilidad.
- Dirigido a objetos.
- Ejecución en dos pasos.
- Seguridad.
- Dinamismo.
- Distributividad.
- Independencia.

Algunas ventajas

1. Es un lenguaje de programación multiplataforma
2. Ofrece una amplia gama de librerías y herramientas
3. Cuenta con un sistema de seguridad incorporado
5. Es un lenguaje de programación de alto nivel
4. Es un lenguaje de programación orientado a objetos

Algunas desventajas

1. Tiene un rendimiento más lento
2. Puede requerir más memoria que otros lenguajes
3. La programación en Java tiene sobrecarga de código

LENGUAJE DE PROGRAMACIÓN. (DART)

Características de Dart

Programación estructurada y flexible. Google diseñó Dart para poder ser utilizado en proyectos de una sola persona hasta proyectos más desarrollados o complejos.

Lenguaje familiar y fácil de aprender. Es un lenguaje realmente sencillo y fácil de aprender. En su sitio web se pueden encontrar varios tutoriales, y también permite colaboraciones de otros desarrolladores.

Permite la adaptación de nueva herramienta a cualquier navegador web. El lenguaje de programación Dart se puede ejecutar de dos maneras; en una máquina virtual (MV), o en un motor de Javascript utilizando un compilador para traducir el código. Esto le permite adaptarse

a cualquier navegador.

Lenguaje basado en clases e interfaces o POO. Gracias a sus basamentos en clases o en la programación orientada a objetos, se facilita la encapsulación y la reutilización del código.

Ventajas de Dart:

Es de acceso gratuito para cualquier persona.

Detrás de su programación se encuentra Google, lo que ofrece perspectivas a largo plazo para el desarrollo del lenguaje.

Dart es fácil de aprender debido a que los desarrolladores han simplificado características complicadas de otros lenguajes.

Funciona en todos los navegadores móviles y de escritorio actuales.

(Fuente: www.hiberus.com)

FRAMEWORK (FLUTTER)

Flutter es una tecnología gratuita y de código abierto que Google desarrolló en mayo de 2017. El objetivo de Flutter es crear aplicaciones nativas para Android e iOS con una única base de código. Este kit de desarrollo móvil se basa en el lenguaje de programación Dart y la arquitectura de programación React, que admite Android Studio, las API de Firebase.

Además, ofrece un desarrollo de aplicaciones de alto rendimiento. Por ello, podemos decir que se trata más bien de un Software Development Kit (SDK) para aplicaciones diseñadas para dispositivos con pantalla táctil, que funcionan bien con iOS y Android.

Pero, ¿por qué Flutter es considerado un SDK, más allá de ser un framework? Pues bien, esto se debe a que contiene todo lo necesario para crear aplicaciones multiplataforma. Esto incluye un motor de renderizado, widgets listos para usar, API de pruebas e integración y herramientas de línea de comandos.

Algunas ventajas:

1. Respuestas inmediatas

- Flutter ofrece la función de recarga en caliente, que permite actualizaciones instantáneas sin necesidad de plugins. Esto también te ayuda a ver las actualizaciones en tiempo real. Si encuentras un error mientras ejecutas el código, el framework te permitirá solucionarlo de inmediato y continuar sin tener que reiniciarlo.

2. Rendimiento nativo

- Otro de los grandes beneficios de utilizar Flutter es que se ejecuta sobre Skia, un motor gráfico que permite un desarrollo rápido y bien optimizado. Asimismo, es indistinguible de las apps nativas, puesto que no depende de intérpretes ni de representaciones de código intermedias.

3. Reducción del tiempo de desarrollo

- Los requisitos para el desarrollo de aplicaciones Flutter son mucho menores. En este

sentido, es seguro decir que el resultado positivo es que no hay gastos de mantenimiento adicionales. Al emplear este kit de desarrollo móvil, es posible crear apps más grandes utilizando características únicas en menos tiempo.

ANDROID STUDIO

Algunas ventajas

Permite ejecutar las compilaciones de forma muy rápida, y así poder comprobar en el momento los fallos y mejoras de la aplicación. Realiza renderizados de layouts en tiempo real, y cuenta con la posibilidad de utilizar parámetros tools. Ejecuta la aplicación en tiempo real desde el propio teléfono móvil. Su potente emulador ayuda a comprobar el estado de la aplicación en el momento, sin necesidad de un ordenador.

Permite simular diferentes dispositivos y tabletas, pudiendo visualizarlas en un mismo entorno. De esta forma podemos trabajar varias aplicaciones simultáneamente y ver las partes de código necesarias de cada una.

Algunas desventajas

Requiere de una gran cantidad de recursos y gasta bastante batería. Así que, para que el emulador trabaje correctamente, necesitarás un buen equipo de trabajo, con gran capacidad de RAM y espacio suficiente en tu disco duro, entre otros requisitos mínimos.

VS CODE

Algunas ventajas

Entre las ventajas de VS Code podemos mencionar las siguientes:

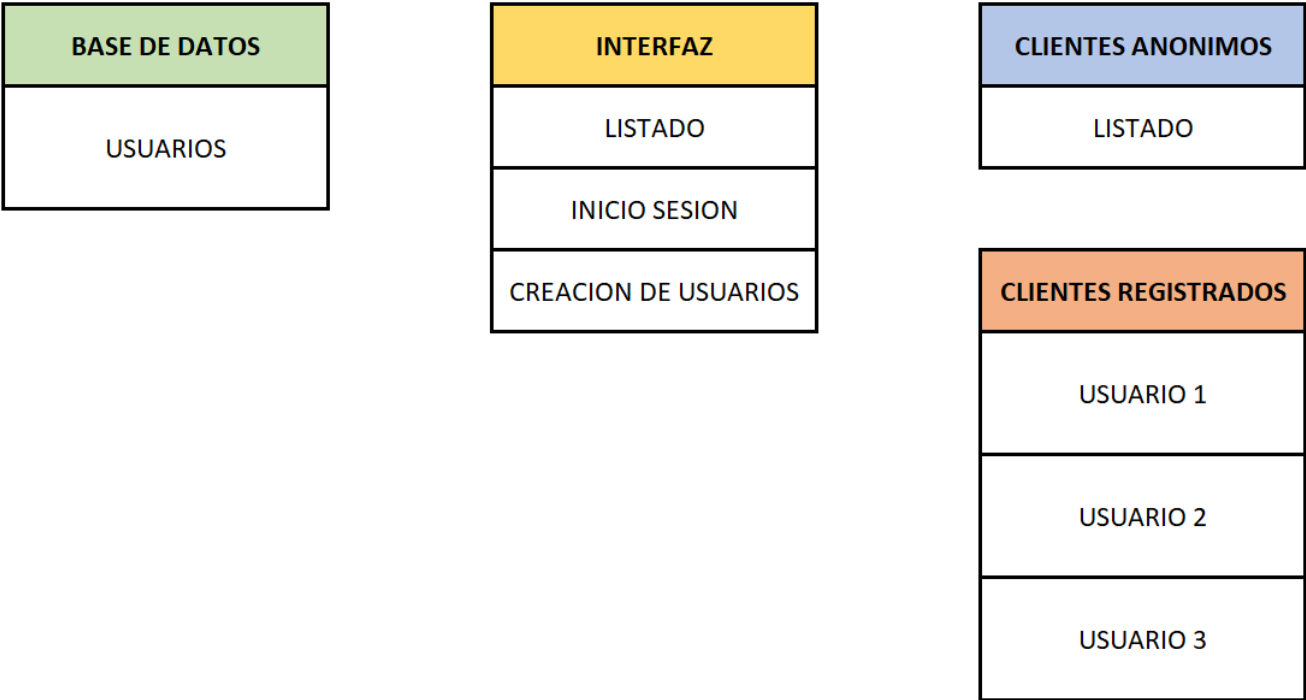
1. Es gratuito y de código abierto
 - VS Code es un software gratuito y de código abierto, lo que significa que no tienes que pagar nada para descargarlo y usarlo. Además, puedes personalizarlo y hacer cambios en el código fuente según tus necesidades.
2. Es altamente personalizable
 - VS Code es altamente personalizable, lo que significa que puedes cambiar la apariencia y la funcionalidad de la herramienta según tus necesidades. Puedes agregar extensiones, temas y atajos de teclado personalizados para hacer tu trabajo más eficiente.
3. Es compatible con varios lenguajes de programación
 - VS Code es compatible con una amplia gama de lenguajes de programación, incluyendo JavaScript, HTML, CSS, Python, Ruby y muchos más. Esto significa que no tienes que cambiar de herramienta cada vez que trabajas con un lenguaje diferente.
4. Tiene una gran comunidad
 - VS Code tiene una gran comunidad de usuarios y desarrolladores que contribuyen con extensiones y temas. Puedes encontrar una gran cantidad de recursos en línea para ayudarte a resolver problemas y aprender más sobre la herramienta.

EQUIPO INFORMÁTICO.

COMPUTADORA PORTÁTIL: ASUS SEPHIROS.

Cuenta con recursos apropiados para el desarrollo de aplicaciones en el IDE Android Studio, pues se sabe que este consume muchos recursos. En este modelo de portátil, corre sin problemas a velocidad consistente y respetable.

DIAGRAMA ARQUITECTONICO DE LA SOLUCION



DESCRIPCIÓN DE LOS COMPONENTES DE LA APLICACIÓN

DIAGRAMA ARQUITECTÓNICO DE LA APLICACIÓN

PETROLIN V2

HOME

LIST

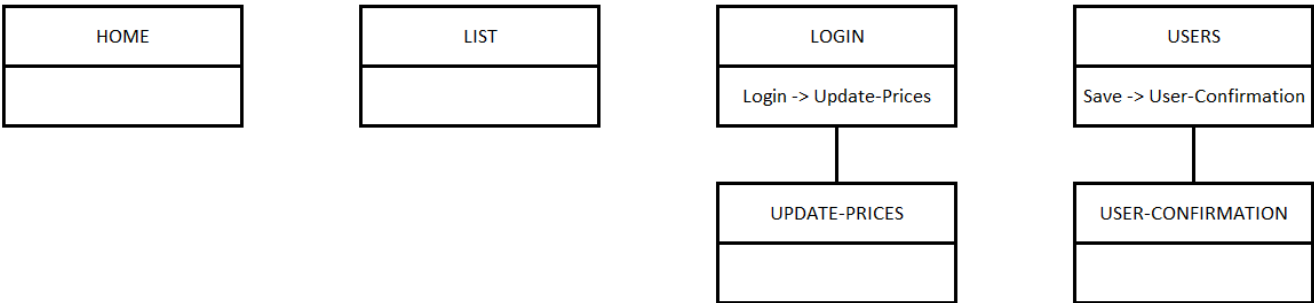
LIST-DETAIL

LOGIN

UPDATE-PRICES

USER

CONFIRMATION-USER-CREATION



DESCRIPCIÓN DE CADA COMPONENTE DE LA SOLUCION

PANTALLA PRINCIPAL (HOME)

BOTÓN HOME

Devuelve a la pantalla principal

BOTÓN LIST

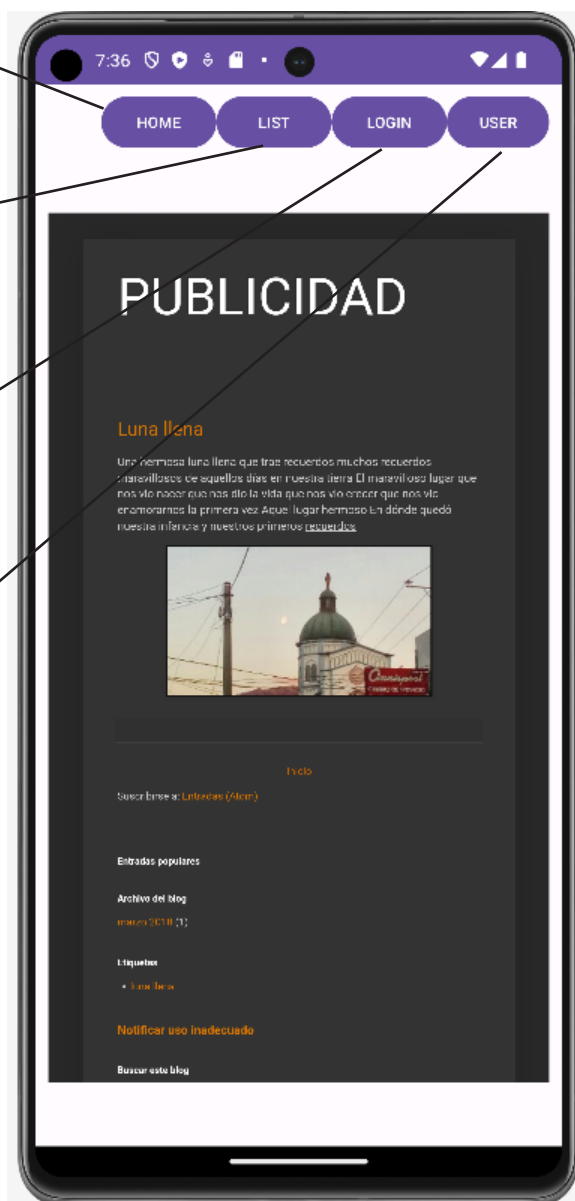
Muestra la pantalla que despliega la lista de estaciones de servicio con los mejores precios

BOTÓN LOGIN

Muestra la pantalla que permite el inicio de sesión a los usuarios registrados (propietarios de gasolineras)

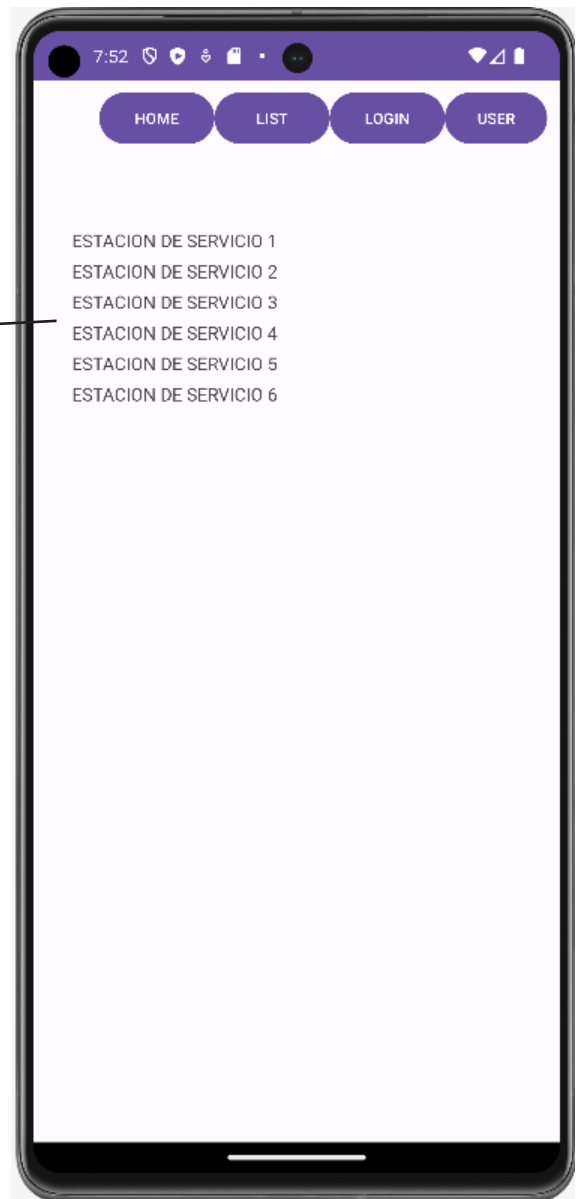
BOTÓN USER

Muestra la pantalla que permite la creación de nuevos usuarios.



PANTALLA LISTA DE ESTACIONES DE SERVICIO (LIST)

Listado de estaciones de servicio



PANTALLA DE INICIO DE SESION (LOGIN)

USER

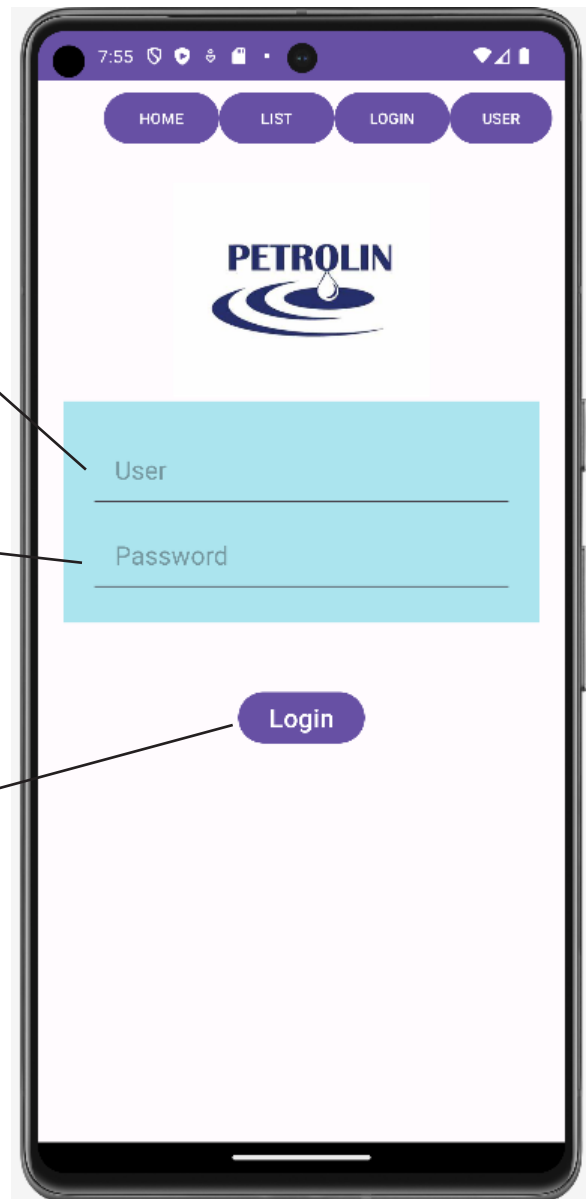
Nombre de usuario, del usuario registrado.

PASSWORD

Contraseña de usuario, del usuario registrado.

BOTON LOGIN

Verifica los credenciales del usuario.
Si la validación es verdadera, envía a la
pantalla de actualización de precios



PANTALLA DE ACTUALIZACION DE PRECIOS (UPDATE-PRICES)

BOTON HOME

Devuelve a la pantalla principal.

BOTON UPDATE

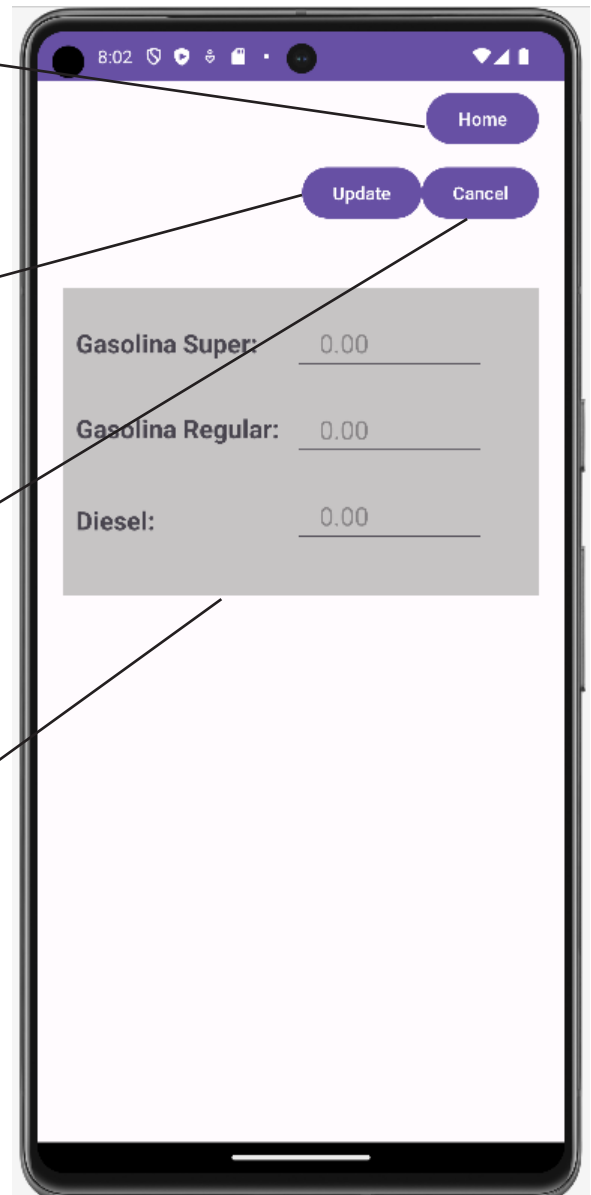
Actualiza los precios de los combustibles.

BOTON CANCEL

Anula la actualizacion de los precios y devuelve a la pantalla LOGIN

PRECIOS DE LOS COMBUSTIBLES

Muestra los precios actuales y permite la edicion para los nuevos precios.



PANTALLA CREACION DE USUARIO (USER-CREATE)

BOTON SAVE

Adiciona un nuevo registro.
Envia a la pantalla Confirmacion de Registro

BOTON CANCEL

Cancela la adicion del registro.
Devuelve a la pantalla principal.

EMPRESA

Nombre de la estacion de servicio.

DIRECCION

Ubicacion de la estacion de servicio.

USUARIO

Nombre de usuario.

CONTRASEÑA

Contraseña de usuario

CODIGO VERIFICACION

Codigo necesario para registrar el nuevo usuario

8:10

HOME LIST LOGIN USER

CREAR USUARIOS

PETROLIN

Save Cancel

Empresa

Direccion

Usuario

Contraseña

Codigo verificacion

CONFIRMACION DE CREACION DE NUEVO USUARIO

BOTON LOGIN

Devuelve a la pantalla LOGIN



CAPITULO III

Estrategia de Implementación de propuesta de solución

PARA IMPLEMENTAR UNA ESTRATEGIA DE APLICACIÓN, SE PUEDEN SEGUIR LOS SIGUIENTES

Investigación acerca de la industria.

La industria de los combustibles para automóviles es muy rentable, razón por la cual existe una enorme competencia.

Aun cuando muchas estaciones están bajo una misma marca, estas no pertenecen a dicha empresa.

Por excelencia, estas, son franquicias. Se entiende entonces, que tantas estaciones existen como propietarios.

La competencia es muy grande, aunque generalmente se encuentran ubicadas geográficamente distantes, asunto que no siempre es así.

Los rangos de precios, por excelencia los controlan las petroleras, sin embargo existe una ventana con la cual los propietarios, pueden hacer sus movimientos para obtener mayores ganancias.

Si se considera que las distancias no suelen ser muy grandes, los consumidores en la mayoría de ocasiones, para abastecerse de combustible, buscan una estación que cumpla con ciertos requisitos, tales como precios y honestidad en el volumen de reciben en concepto de galones adquiridos.

En vista de tal competencia por los clientes, estos siempre procuran un rango de precios, acorde a la ubicación geográfica, considerando la posibilidad, que los clientes busquen o no otra estación en las cercanías que les ofrezca mejores precios.

Definir el usuario objetivo.

Existen diferentes industrias que consumen combustibles fósiles, para realizar sus actividades diarias, pero nuestro interés estriba en los conductores de automotores.

Existe un parque vehicular, considerablemente alto y una enorme mayoría se desplaza por el gran San Salvador, todos los días.

Esta situación les presenta acceso a estaciones de servicio en diferentes ubicaciones geográficas y como es de esperar, siempre buscan una estación que les ofrezca los mejores precios.

Procurar los mejores precios, aún cuando sea de tan solo centavos, es una tarea diaria, pues todo conductor sabe, que un centavo ahorrado por día, representa un valor considerable en el largo plazo.

Identificar a los principales competidores.

Hasta el presente día, no existe una aplicación que muestre los precios de los combustibles en las diferentes estaciones de servicio del área metropolitana de San Salvador.

Existen aplicaciones que prestan otro tipo de servicios, como puntos leal, en la cual se registran puntos por consumo de una misma marca, sin importar la ubicación de la estación, pero esta no presenta los precios al día.

En consecuencia, esta vez no se cumple aquella premisa, de “No eres el primero”, pues a razón de los hechos, PETROL IN, será la primera aplicación que muestre los precios de combustibles de la diferentes estaciones de servicios del área metropolitana de San Salvador.

Diseñar una página web oficial de la aplicación.

Es muy conveniente la creación de una página web para la aplicación, considerando que tanto consumidores como propietarios de estaciones querrán información que de otra forma, sería muy tedioso ofrecer; por teléfono por ejemplo.

Se ha considerado este hecho, pues es necesario disponer información relevante a los usuarios que esté disponible 24/7

Aun no se ha iniciado el proceso de diseño, pero es parte de los planes de desarrollo.

Aprovechar las redes sociales para lucir las funcionalidades de la aplicación.

Hoy en día, las redes sociales representan una herramienta valiosa para el marketing.

El alcance que estas tienen es considerablemente alto y es de tomar en cuenta que se puede realizar campañas de marketing, tanto gratuitas como de paga.

Por supuesto las campañas de paga, poseen un alcance mucho mayor y existen diferentes planes que se ajustan al presupuesto.

Crear una estrategia de marketing de contenidos.

El tema de contenidos es bastante delicado, pues no suelen tener el impacto que se desea, si los contenidos se presentan en redes sociales, es de tomar y muy en cuenta, que estos no se reproducen automáticamente aún cuando si se puede, la mayoría de usuarios configuran sus dispositivos para evitar la reproducción automática, con la finalidad de ahorrar en el consumo de datos.

En el caso de plataformas como youtube, para que estos contenidos sean visualizados por los usuarios, es imperante que se presenten en campañas publicitarias de paga y la duración idealmente no debe superar los siete segundos.

Porque siete segundos, pues es la ventana que youtube ofrece para poder detener la

reproducción de los anuncios.

Y debe notarse, que más del 90% de usuarios detienen la reproducción de anuncios en ese tiempo.

Por tanto. Debe crearse contenidos con una duración de siete segundos y desplegarlos en una campaña de paga segmentada para garantizar la visualización de los mismos.

Optimizar la vista de la aplicación en las tiendas de aplicaciones.

La optimización en las tiendas de aplicaciones, es de suma importancia, pues esto garantiza la disponibilidad para los usuarios, se debe tomar en cuenta algunos aspectos, tales como:

TITULO:

El título es importante, pues el nombre que disponga una aplicación debe ser fácil de recordar y significativo.

El título no se puede elegir al azar, este debe representar el servicio que presta, también es de tomar en cuenta que el nombre de una aplicación se puede cambiar durante la marcha, este no es rígido y por lo mismo, debe tomarse el tiempo necesario para asignar un título significativo.

DESCRIPCIÓN

La descripción es otro factor clave, pues esta debe presentar al usuario en unas pocas líneas el cometido de la aplicación y convencerlo de descargar y utilizar la aplicación.

Es importante utilizar palabras técnicas que tienda a confundir al usuario. Es necesario considerar la gama de posibles usuarios.

PALABRAS CLAVE

Que una aplicación se encontrada fácilmente, depende en gran medida de las palabras clave que se utilicen.

Es normal que el usuario utilice palabras relacionadas con el tipo de servicio que busca y no precisamente por el nombre de la aplicación, pues en muchas ocasiones desconoce el nombre.

En consecuencia, la adecuada utilización de palabras clave, facilitará la búsqueda de los potenciales usuarios.

Existen más datos a tomar en cuenta, pero estos son los más relevantes.

Lanzar campañas pagadas en social media y buscadores.

En El Salvador, el buscador por excelencia es Google.

Se realizará una campaña de posicionamiento en este buscador. Para tal efecto se cuenta con un especialista en SEO.

Igualmente existen dos tipos de campaña: Gratuita y de paga.

Es de considerar inicialmente una campaña gratuita, para su pre-lanzamiento y posteriormente una campaña de paga el lanzamiento.

Estas campañas también son muy económicas, y existen varios planes a considerar, como por ejemplo, paga por clic.

Solo se paga por los clic efectivos y suele ser al menos al inicio, mucho más cómodo que una paga por periodos.

PRESUPUESTO DE LA IMPLEMENTACION (EN DOLARES USA)

RUBRO	TIEMPO / HORAS	VALOR / HORA	TOTAL
Investigación acerca de la industria.	4	\$ 20.00	\$ 80.00
Definir el usuario objetivo.	1	\$ 20.00	\$ 20.00
Identificar a los principales competidores.	1	\$ 20.00	\$ 20.00
Diseño plan de mercadeo en redes sociales	5	\$ 30.00	\$ 150.00
Crear una estrategia de marketing de contenidos.	5	\$ 30.00	\$ 150.00
Optimizar la vista de la aplicación en las tiendas de aplicaciones.	5	\$ 20.00	\$ 100.00
Lanzar campañas pagadas en social media y buscadores.	5	\$ 40.00	\$ 200.00
TOTAL			\$ 720.00

ANÁLISIS DE RESULTADOS

En el mercado existen diferentes tipos de consumidores, situación que dificulta o facilita la posición de cualquier producto en el mercado.

Sin embargo, es de notar, que cuando un servicio aporta valor a las actividades diarias de un usuario, puesto servicio tiene una alta aceptación.

Tómese en cuenta que las campañas de posicionamiento, necesariamente cumplen su cometido.

Existen muchos consumidores, que no sienten gusto alguno por la publicidad y procuran evitarla, siempre que sea posible.

Existen técnicas que aun cuando el usuario no gusta de la publicidad, este, siempre se ve obligado a observarla.

Quizá parezca algo no ético, pero se debe tomar en cuenta que la publicidad es de vital importancia en el mundo de los negocios.

Por esta razón los estudiosos del mercadeo, han descubierto algunos tips, que ayudan para que la publicidad sea servida al usuario.

Se conoce, que aun cuando el usuario no gusta de la publicidad, esta siempre cumple su propósito.

Redes sociales.

Observese la imagen al calce de la pagina.

Corresponde a una campaña en redes sociales, de no paga.

La diferencia con la campañas de paga, es el lugar de la pagina en donde se muestra el arte.

En las campañas de paga, el arte tiene una posición que siempre se espeta, pero no es exclusivo, ya que otros usuarios también han pagado por mostrar su publicidad, pero tiene la ventaja que se filtra por segmentos según el tipo de usuario al que se desea llegar.

En la campañas de no paga, esta de coloca en los perfiles del usuario, y se sabe a ciencia cierta que esta se mueve constantemente hacia abajo, según la dinámica de los usuarios.

Youtube.

En youtube, es muy diferente.

Una de las formas de colocar publicidad de no paga, es editando contenido que la incluya.

Puesto que por defecto la única publicidad que muestra en campañas de paga.

Con la salvedad que el usuario puede omitir cualquier pauta publicitaria después de 7 segundos.



Considerando este margen de tiempo, se aconseja, que el video no sobrepase este tiempo.

Todo video que se enmarque en el margen de los 7 segundos, indefectiblemente será visto por los consumidores.

Posicionamiento en Google Search.

Google Search, dispone de un servicio, que inicialmente es gratis. Al alcanzar cierto numero de clics efectivos, se inicia la facturación.

Cuando un usuario realiza búsquedas de cualquier tipo, la publicidad se muestra.

La ciencia estriba en las palabras clave que se utilicen.

Ciertamente, estamos hablando de un universo demasiado grande, y lo que se busca es, que cuando un usuario realice búsquedas de servicios o productos similares, entonces se muestre la publicidad. No se puede mostrar productos para niños a un usuario que busca partes para una computadora, pues está claro, que no se percatará de la misma, por estar fuera de su interés en el momento.

Es altamente efectivo, siempre y cuando se utilicen palabras clave adecuadas.

Resultados

En general los resultados son muy buenos, no se ha alcanzado la excelencia, dado que el tiempo transcurrido, no es el adecuado.

Sin embargo, en el tiempo, se ha alcanzado una aceptación nada despreciable.

CONCLUSIONES

En la actualidad, un muy alto porcentaje de personas, utiliza su móvil para realizar actividades que de otra forma, tendría que acercarse a instituciones que le resuelvan la necesidad en cuestión.

Dígase de tramites bancarios, y una infinidad de servicios.

Los móviles, prestan servicio de alto valor para los usuarios, especialmente si hablamos de información.

Como es lógico pensar, todos queremos estar informados con respecto a aquellos tópicos que nos interesan.

Puede ser noticias internacionales o bien nacionales.

Uno de los aspectos, que todo usuario de automóviles, necesita conocer, es el rango de precios de los combustibles que consume.

Existen varios tipos de conductores, a saber, los que conducen a diario y durante las horas laborales, porque su ocupación así lo exige. El consumo en combustibles de estos usuarios, es considerable y su desplazamiento también.

Razón por la que les beneficia conocer los mejores precios disponibles en el área donde se encuentren en un momento dado del día.

Aplicaciones, que proporcionen información, como es el caso de PETROLIN, siempre serán bien aceptadas por el público.

RECOMENDACIONES

Para desarrollar aplicaciones móviles, no existen muchas alternativas.

Muy pocos móviles disponen de Windows Mobile, la gran mayoría de usuarios utilizan dispositivos con Android OS o bien IOs.

Esta situación limita mucho las posibilidades.

Si bien existen algunos frameworks, que los más versados en el tema sostienen que facilita el desarrollo de aplicaciones, puede mencionarse React Native, Node Js, Xamarin, Flutter y otros más.

Cada framework, tiene sus cualidades, su forma particular de manejar las posibilidades de una aplicación móvil.

En el caso de Android, yo, lo considero muy engorroso, para mostrar una simple herramienta como un menú, se requiere una enorme cantidad de código, situación que en otros lenguajes se resuelve con una simple instrucción.

Considerando lo anterior y que no existen alternativas, mas que un sistema engorros y burocrático para el desarrollo móvil, es recomendable seleccionar un framework. El que más agrade y procurar profundizar cuanto sea posible en su lógica de negocio, para obtener el mayor provecho posible.

Si se toma en cuenta que cada framework, tiene su propia burocracia, es muy engorroso procurar dominar dos o tres de estos.

Se recomienda encarecidamente enfocarse en uno solo, el de mayor atractivo, pero solo uno.

De esta forma. Podrá aprovecharse la riqueza que cada uno aporta.

LENGUAJE DART

Fuente: www.hiberus.com

FLUTTER

(Fuente: <https://www.grupoebim.com/blog/ventajas-desventajas-flutter/>)

LENGUAJE JAVA

<https://blog.hubspot.es/website/ventajas-desventajas-java>

ANDROID STUDIO

(Fuente: <https://scoreapps.com/>)

VS CODE

(Fuente: <https://webdesigncusco.com/>)

ANEXOS

PANTALLA PRINCIPAL

```

package com.ten.firbasedemo;

import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.webkit.WebView;
import android.webkit.WebViewClient;
import android.widget.EditText;
import android.widget.ListView;
import android.widget.Toast;

import androidx.appcompat.app.AppCompatActivity;

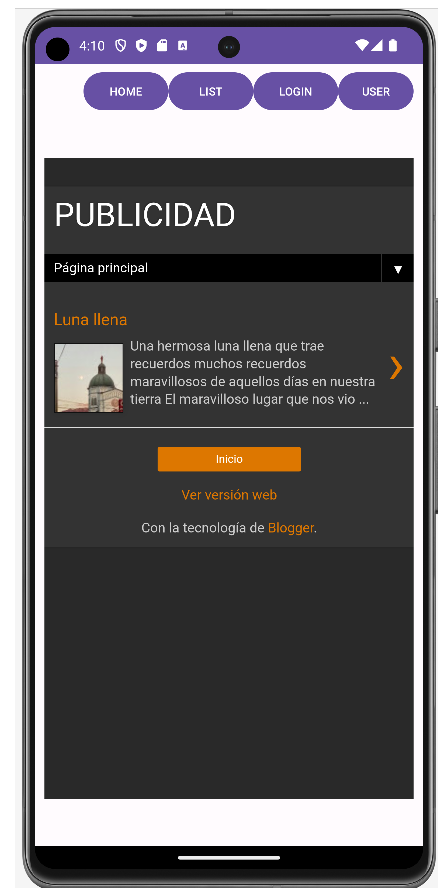
import com.google.firebase.FirebaseApp;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;

import java.util.UUID;

public class MainActivity extends AppCompatActivity {
    public EditText txtNombre;
    public EditText txtUsuario;
    public EditText txtPass;
    public ListView lstUsers;

    //Referencia a base de datos Firebase
    FirebaseDatabase fbDatabase;
    DatabaseReference dbReference;

```



```

@Override

protected void onCreate(Bundle savedInstanceState) {

    super.onCreate(savedInstanceState);

    setContentView(R.layout.activity_main);


    WebView webview = (WebView) findViewById(R.id.webview);

    webview.getSettings().setJavaScriptEnabled(true);

    webview.getSettings().setBuiltInZoomControls(true);


    webview.loadUrl("https://hortizja.blogspot.com/");

    webview.setWebViewClient(new WebViewClient(){

        public boolean shouldOverrideUrlLoading(WebView webview, String url){

            return false;

        }

    });


    IniciarFirebase();

}


public void IniciarFirebase(){

    FirebaseApp.initializeApp(this);

    fbDatabase = FirebaseDatabase.getInstance();

    dbReference = FirebaseDatabase.getInstance().getReference();

}


public void Main_IniciarHome(View view){

    Intent home = new Intent(this, MainActivity.class);

    startActivity(home);

```

```
}
```

```
public void Main_IniciarListado(View view){  
    Intent listado = new Intent(this, Listado.class);  
    startActivity(listado);  
}
```

```
public void Main_IniciarLogin(View view){  
    Intent login = new Intent(this, Login.class);  
    startActivity(login);  
}
```

```
public void Main_IniciarCrearUsuarios(View view){  
    Intent UsuariosCrear = new Intent(this, Usuarios_Crear.class);  
    startActivity(UsuariosCrear);  
}  
}
```

LISTA

```

package com.ten.firbasedemo;

import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.os.Bundle;
import android.view.View;

public class Listado extends AppCompatActivity {

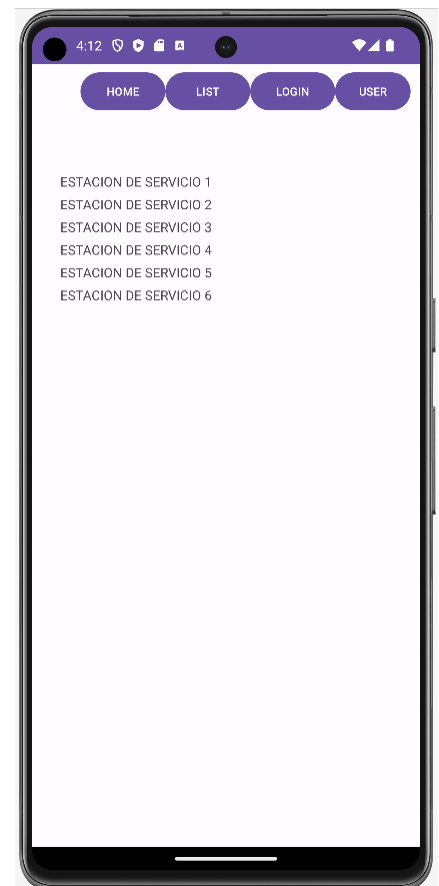
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_listado);
    }

    public void Listado_IniciarHome(View view){
        Intent home = new Intent(this, MainActivity.class);
        startActivity(home);
    }

    public void Listado_IniciarListado(View view){
        Intent listado = new Intent(this, Listado.class);
        startActivity(listado);
    }

    public void Listado_IniciarLogin(View view){
        Intent login = new Intent(this, Login.class);

```



```
        startActivity(login);  
    }  
    public void Listado_IniciarCrearUsuarios(View view){  
        Intent UsuariosCrear = new Intent(this, Usuarios_Crear.class);  
        startActivity(UsuariosCrear);  
    }
```

LOGIN

```

package com.ten.firbasedemo;

import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.os.Bundle;
import android.view.View;

public class Login extends AppCompatActivity {

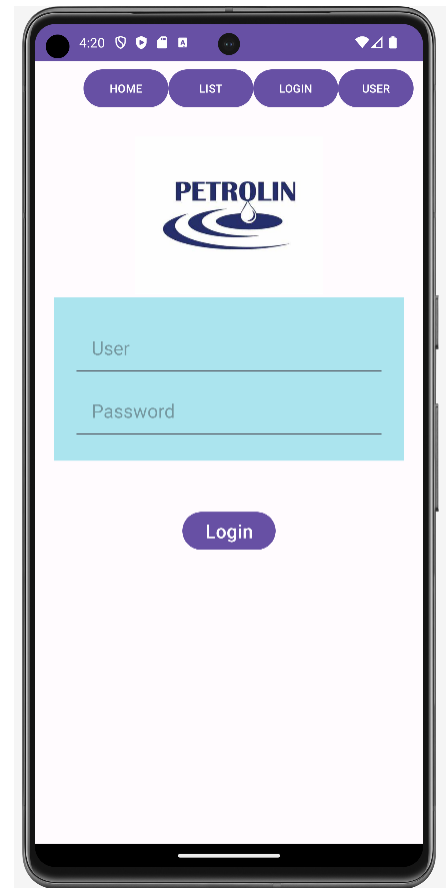
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_login);
    }

    //Actividades
    public void Login_IniciarHome(View view){
        Intent home = new Intent(this, MainActivity.class);
        startActivity(home);
    }

    public void Login_IniciarListado(View view){
        Intent listado = new Intent(this, Listado.class);
        startActivity(listado);
    }

    public void Login_IniciarLogin(View view){

```



```

        Intent login = new Intent(this, Login.class);
        startActivity(login);
    }

    public void Login_IniciarCrearUsuarios(View view){
        Intent UsuariosCrear = new Intent(this, Usuarios_Crear.class);
        startActivity(UsuariosCrear);
    }

    //Inicio de sesion
    public void Login_IniciarSesion(View view){
        try {

            Modificar_Precios();
        } catch (Exception e){

        }

    }

    public void Modificar_Precios(){
        Intent mp = new Intent(this, Usuario_Modificar_Precios.class);
        startActivity(mp);
    }

}

```

MODIFICAR PRECIOS

```

package com.ten.firbasedemo;

import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.os.Bundle;
import android.view.View;

public class Usuario_Modificar_Precios extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_usuario_modificar_precios);
    }

    public void Actualizar_Precios(View view){
        try{

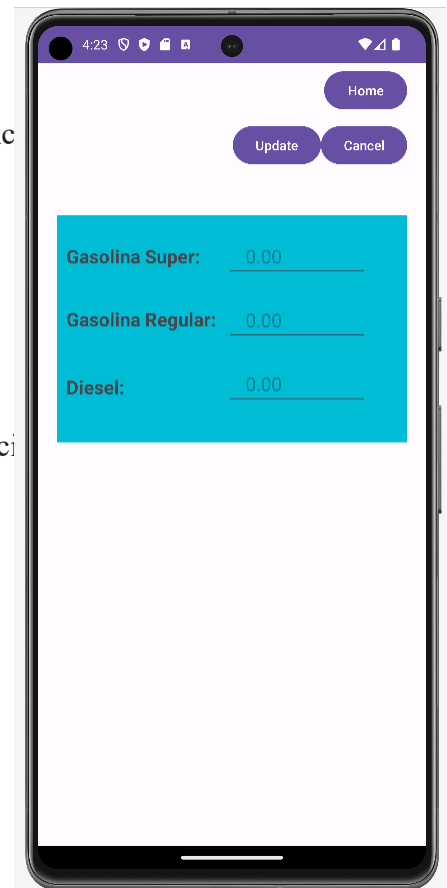
        }catch (Exception e){

        }

    }

    public void Modificar_Home(View view){
        Intent Home = new Intent(this, MainActivity.class);
        startActivity(Home);
    }
}

```



```
public void Cancelar_Update(View view){  
    Intent Home = new Intent(this, Login.class);  
    startActivity(Home);  
}  
  
}
```

CREAR USUARIOS

```

package com.ten.firbasedemo;

import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.EditText;
import android.widget.Toast;

import androidx.appcompat.app.AppCompatActivity;

import com.google.firebase.FirebaseApp;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;

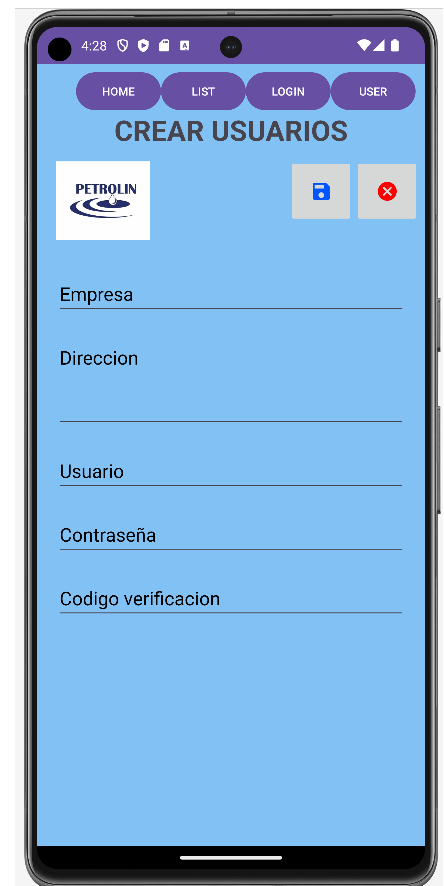
import java.util.UUID;

public class Usuarios_Crear extends AppCompatActivity {

    public EditText txtNombre;
    public EditText txtDireccion;
    public EditText txtUsuario;
    public EditText txtPass;
    public EditText txtVerificacion;
    public EditText txtCsuper;
    public EditText txtCregular;
    public EditText txtCdiesel;

    //Referencia a base de datos Firebase
    FirebaseDatabase fbDatabase;

```



```
DatabaseReference dbReference;
```

```
@Override
```

```
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_usuario_crear);

    txtNombre = findViewById(R.id.edtNombre);
    txtDireccion = findViewById(R.id.edtDireccion);
    txtUsuario = findViewById(R.id.edtUsuario);
    txtPass = findViewById(R.id.edtPass);
    txtVerificacion = findViewById(R.id.edtVerificacion);
    txtCsuper = findViewById(R.id.edtCsuper);
    txtCregular = findViewById(R.id.edtCregular);
    txtCdiesel = findViewById(R.id.edtCdiesel);
```

```
    IniciarFirebase();
```

```
}
```

```
public void Add(View view){
    try{
        ValidarCajas();
    }catch (Exception e){
        Toast.makeText(this, "Error fatal", Toast.LENGTH_SHORT).show();
    }
}
```

```
public void LimpiarC(View view){
```

```

LimpiarCajas();

Intent Home = new Intent(this, MainActivity.class);
startActivity(Home);
}

public void LimpiarCajas(){
    txtNombre.setText("");
    txtDireccion.setText("");
    txtUsuario.setText("");
    txtPass.setText("");
    txtVerificacion.setText("");
}

public void IniciarFirebase(){
    FirebaseApp.initializeApp(this);
    fbDatabase = FirebaseDatabase.getInstance();
    dbReference = FirebaseDatabase.getInstance().getReference();
}

public void InsertarUsuario(){
    try{
        String Nombre = txtNombre.getText().toString();
        String Direccion = txtDireccion.getText().toString();
        String Usuario = txtUsuario.getText().toString();
        String Pass = txtPass.getText().toString();
        String Verificacion = txtVerificacion.getText().toString();
        String Csuper = txtCsuper.getText().toString();
        String Cregular = txtCregular.getText().toString();
        String Cdiesel = txtCdiesel.getText().toString();
    }
}

```

```

        Users user = new Users();
        user.setUid(UUID.randomUUID().toString());
        user.setName(Nombre);
        user.setDireccion(Direccion);
        user.setUser(Usuario);
        user.setPass(Pass);
        user.setVerificacion(Verificacion);
        //
        user.setCsuper(Csuper);
        user.setCregular(Cregular);
        user.setCdiesel(Cdiesel);
        //
        dbReference.child("Users").child(user.getUid()).setValue(user);
    } catch (Exception e){
        Toast.makeText(this, "Error fatal",Toast.LENGTH_LONG).show();
    }

    LimpiarCajas();

    Intent Crear_Exito = new Intent(this, Usuario_Creado_Exito.class);
    startActivity(Crear_Exito);

}

public void ValidarCajas(){
    if(txtNombre.getText().toString().length() == 0) {
        txtNombre.setError("Nombre requerido!");
        //Toast.makeText(this, "Campo requerido",Toast.LENGTH_LONG).show();
    }
}

```

```

    }else {InsertarUsuario();}

    if(txtDireccion.getText().toString().length() == 0 ){
        txtDireccion.setError("Direccion requerida!");
        //Toast.makeText(this, "Campo requerido",Toast.LENGTH_LONG).show();
    }else {InsertarUsuario();}

    if(txtUsuario.getText().toString().length() == 0 ){
        txtUsuario.setError("Usuario requerido!");
        //Toast.makeText(this, "Campo requerido",Toast.LENGTH_LONG).show();
    }else {InsertarUsuario();}

    if(txtPass.getText().toString().length() == 0 ){
        txtPass.setError("Password requerido!");
        //Toast.makeText(this, "Campo requerido",Toast.LENGTH_LONG).show();
    }else {InsertarUsuario();}

    if(txtVerificacion.getText().toString().length() == 0 ){
        txtVerificacion.setError("Codigo verificacion requerido!");
        //Toast.makeText(this, "Campo requerido",Toast.LENGTH_LONG).show();
    }else {InsertarUsuario();}

}

//Iniciar actividades
public void Usuarios_IniciarHome(View view){
    Intent home = new Intent(this, MainActivity.class);
    startActivity(home);
}

```

```
public void Usuarios_IniciarListado(View view){  
    Intent listado = new Intent(this, Listado.class);  
    startActivity(listado);  
}  
  
public void Usuarios_IniciarLogin(View view){  
    Intent login = new Intent(this, Login.class);  
    startActivity(login);  
}  
public void Usuarios_IniciarCrearUsuarios(View view){  
    Intent UsuariosCrear = new Intent(this, Usuarios_Crear.class);  
    startActivity(UsuariosCrear);  
}  
  
}
```

USERS (CLASS)

```
package com.ten.firbasedemo;

public class Users {

    public String uid;
    public String name;
    public String direccion;
    public String user;
    public String pass;
    public String verificacion;

    public String csuper;
    public String cregular;
    public String cdiesel;
    public Users() {
    }

    public String getUid() {
        return uid;
    }

    public void setUid(String uid) {
        this.uid = uid;
    }

    public String getName() {
        return name;
    }
}
```

```
public void setName(String name) {  
    this.name = name;  
}
```

```
public String getDireccion() {  
    return direccion;  
}
```

```
public void setDireccion(String direccion) {  
    this.direccion = direccion;  
}
```

```
public String getUser() {  
    return user;  
}
```

```
public void setUser(String user) {  
    this.user = user;  
}
```

```
public String getPass() {  
    return pass;  
}
```

```
public void setPass(String pass) {  
    this.pass = pass;  
}
```

```
public String getVerificacion() {
```

```
        return verificacion;
    }

    public void setVerificacion(String verificacion) {
        this.verificacion = verificacion;
    }

    public String getCsuper() {
        return csuper;
    }

    public void setCsuper(String csuper) {
        this.csuper = csuper;
    }

    public String getCregular() {
        return cregular;
    }

    public void setCregular(String cregular) {
        this.cregular = cregular;
    }

    public String getCdiesel() {
        return cdiesel;
    }

    public void setCdiesel(String cdiesel) {
        this.cdiesel = cdiesel;
    }
}
```

```
}

@Override
public String toString() {
    return user;
}
}
```