INSTITUTO DE EDUCACIÓN SUPERIOR TECNOLÓGICO PRIVADO CERTUS.



**EVALUACIÓN DE LA ACTIVIDAD DE APRENDIZAJE 3:**

**EVIDENCIA 3**

**CICLO: III**

**GRUPO: 06**

**DOCENTE:**

EDUARDO ANTONIO BARDALES BRUNO

**Presentado por:**

Gutiérrez Díaz, Maryhan

Flores Jacome, Angelo

Zapata Valencia, Diego

Condori Sihuayro, Anthony

Oyola Palomino, Antonella

**2023**

**ÍNDICE**

[**INTRODUCCIÓN 3**](#_u2rusca87bvm)

[**HISTORIA DE LA EMPRESA 4**](#_53f8md97w89r)

[**CARACTERÍSTICAS DE LA EMPRESA 5**](#_pdwv1j3ei36k)

[MISIÓN 5](#_knisbqt5tw51)

[VISIÓN 5](#_xmelwmjywd30)

[**OBJETIVOS 6**](#_uxj2qdecetzj)

[**REQUERIMIENTOS 7**](#_6y659yveah7k)

[REQUERIMIENTOS FUNCIONALES 7](#_hh39adpt9ir9)

[REQUERIMIENTOS NO FUNCIONALES 8](#_7fhi0ui7h370)

[**ANÁLISIS Y DISEÑO 9**](#_e5q0r1k01wp9)

[INTERFAZ DE USUARIO 9](#_tegmp35v4rhm)

[INTERFAZ DE USUARIO 10](#_9o7b0rlodoq6)

[CATÁLOGOS DE PRODUCTOS 11](#_zd4un6ag98zu)

[GEOLOCALIZACIÓN 11](#_k1kc0amsjrix)

[PROCESO DE PAGO 12](#_jlpw1qm1fmsg)

[BARRA DE NAVEGACIÓN 12](#_eth8dty8ohr1)

[**DIAGRAMA DE CASOS DE USOS 13**](#_ckm4yy38c0dg)

[**DIAGRAMA DE CASOS DE SECUENCIA 14**](#_tcd461pxgkjy)

[**CONCLUSIONES Y RECOMENDACIONES 15**](#_2qx3n0frgms2)

# **INTRODUCCIÓN**

En el siguiente informe tiene como objetivo presentar la propuesta de desarrollo de una aplicación móvil para la empresa Distribuidora Navarrete. La distribuidora Navarrete es una compañía especializada en la distribución de productos escolares u oficinas, artículos de limpieza e higiene personal , que busca mejorar su presencia en el mercado mediante la creación de una aplicación móvil. La aplicación móvil permitirá a los clientes acceder fácilmente al catálogo de productos de la empresa, realizar pedidos en línea, conocer promociones y tener una experiencia de usuario más eficiente y satisfactoria. Este informe analizará las principales características y beneficios de la aplicación, así como la viabilidad técnica y financiera del proyecto.

# **HISTORIA DE LA EMPRESA**

La empresa Distribuidora Navarrete S.A. es una empresa peruana que se dedica a la distribución de productos de consumo masivo en el mercado peruano. Fue fundada en 1990 por los hermanos Navarrete, quienes iniciaron sus operaciones con la distribución de bebidas y productos de limpieza.

En sus primeros años, la empresa se enfocó en el mercado local y se expandió en la ciudad de Lima y sus alrededores. Con el tiempo, la empresa logró consolidarse en el mercado peruano y comenzó a expandirse a nivel nacional.

En la década de 2000, Distribuidora Navarrete S.A. Se convirtió en una de las principales empresas de distribución en el mercado peruano, gracias a su amplia red de distribución y su capacidad para llegar a los puntos más remotos del país.

En la actualidad, la empresa cuenta con una amplia gama de productos en su portafolio, que incluye bebidas, productos de limpieza, alimentos, productos de cuidado personal, entre otros. Además, ha ampliado su área de distribución, llegando a cubrir todo el territorio peruano.

Distribuidora Navarrete S.A. Ha logrado consolidarse como una empresa líder en la distribución de productos de consumo masivo en el mercado peruano, gracias a su compromiso con la calidad, la eficiencia en la distribución y el servicio al cliente.

# **CARACTERÍSTICAS DE LA EMPRESA**

**EMPRESA SELECCIONADA:** DISTRIBUIDORA NAVARRETE

**RUC:** 20100030676

**RAZÓN SOCIAL:** DISTRIBUIDORA NAVARRETE S.A

**TIPO EMPRESA:** SOCIEDAD ANÓNIMA

**CONDICIÓN:** ACTIVO

**DIRECCIÓN ACTUAL:** CERCADO DE LIMA AV.NICOLAS DE PIEROLA NRO.1463

**DISTRITO/CIUDAD:** LIMA

**FECHA DE INICIO DE ACTIVIDADES:** 19 DE MARZO DEL 1980

**CIIU:** 51906

## **MISIÓN**

Proporcionar a nuestros clientes la mejor experiencia de compra, ofreciendo una amplia gama de productos de calidad a precios competitivos y con un servicio excepcional. Buscamos alcanzar el liderazgo en el mercado y ser reconocidos por nuestro compromiso con la satisfacción del cliente, la innovación y la responsabilidad social.

### **VISIÓN**

Convertirnos en la empresa líder en distribución de productos de consumo en el mercado nacional. Queremos ser reconocidos como la mejor opción para nuestros clientes y proveedores, y ser una empresa respetada y admirada por nuestra ética empresarial, nuestra contribución al desarrollo socioeconómico sostenible y nuestro compromiso con el bienestar de la sociedad en general.

# **OBJETIVOS**

**OBJETIVOS GENERALES**

Desarrollar una aplicación móvil para la Distribuidora Navarrete que proporcione una experiencia de usuario intuitiva y atractiva.

**OBJETIVOS ESPECÍFICOS**

* Definir las características y funcionalidades de alta prioridad de la aplicación para satisfacer las necesidades y expectativas de los clientes.
* Diseñar una interfaz de usuario intuitiva y vistosa para que la aplicación refleje la imagen y valores de la empresa.
* Integrar la aplicación con los sistemas de gestión de la empresa (inventario, facturación, envío, etc).
* Realizar pruebas precisas de la aplicación para garantizar su estabilidad, rendimiento y usabilidad.
* Analizar y monitorear el rendimiento de la aplicación mediante análisis de datos para identificar áreas de mejora y oportunidades de crecimiento.

# **REQUERIMIENTOS**

## **REQUERIMIENTOS FUNCIONALES**

* **Registros de usuarios:** Permitir a los usuarios crear una cuenta en la aplicación para realizar compras y recibir actualizaciones de promociones y ofertas.
* **Catálogo de productos:** Mostrar en la aplicación el catálogo completo de productos escolares y de oficina disponibles para su compra.
* **Métodos de pago:** La aplicación móvil debe permitir a los usuarios pagar por los productos seleccionados utilizando diferentes métodos de pago, como tarjeta de crédito, débito, yape, entre otros.
* **Geolocalización:** La aplicación móvil debe permitir a los usuarios buscar la ubicación de la tienda física más cercana.
* **Promociones y Descuentos:** La aplicación móvil debe mostrar promociones y descuentos especiales para los productos.
* **Enlazado a redes sociales:** La aplicación móvil debe permitir a los usuarios compartir sus productos favoritos y promociones en sus redes sociales para aumentar la relevancia social de la empresa.
* **Calificaciones y comentarios:** La aplicación móvil debe permitir a los usuarios calificar y comentar sobre los productos que han adquirido.
* **Barra de navegación:** La aplicación móvil debe permitir a los usuarios desplazarse fácilmente por las secciones de la aplicación.
* **Paleta de colores:** La aplicación móvil contará con una paleta de colores vistosa, los colores seleccionados son rojo, amarillo y blanco. Caracterizando a la empresa.
* **Tamaño y estilo de fuente:** Para aumentar la vistosidad de nuestra aplicación, seleccionamos diferentes tamaños de fuente de Nº25 a Nº80. Para el estilo de fuente se importará el paquete de GoogleFonts y se seleccionará la fuente ConcertOne.
* **Botones de redirección:** La aplicación móvil contará con botones de redirección para que el usuario pueda navegar por la aplicación rápidamente.

### **REQUERIMIENTOS NO FUNCIONALES**

* **Interfaz de usuario:** El interfaz de usuario debe ser intuitivo y fácil de usar para asegurar una buena experiencia de usuario.
* **Seguridad:** La aplicación móvil debe ser segura y proteger la información personal y financiera del usuario.
* **Rendimiento:** La aplicación móvil debe ser rápida y eficiente, lo que significa que debe tener tiempos de carga rápidos y no tener retrasos durante el proceso de pago.
* **Usabilidad:** La aplicación móvil debe ser fácil de usar y explorar, incluso para los usuarios que no estén familiarizados con la tecnología.
* **Diseño adaptable:** La aplicación móvil debe ser compatible con cualquier tipo de dispositivo móviles y tamaños de pantalla.
* **Escalabilidad:** La aplicación móvil debe ser escalable para poder manejar un creciente número de usuarios y transacciones.
* **Mantenibilidad**: La aplicación móvil debe ser fácil de mantener y actualizar, para lo cual se debe seguir buenas prácticas de programación y documentación.
* **Disponibilidad:** La aplicación móvil debe estar disponible en todo momento para que los usuarios puedan realizar sus compras en cualquier momento.

# **ANÁLISIS Y DISEÑO**

Para la creación del código fuente de nuestra aplicación móvil se realizó previamente un prototipo con el contenido seleccionado minuciosamente por nuestros integrantes. Este prototipo se irá actualizando con nuevas ideas y mejoras.

## 

## **INTERFAZ DE USUARIO**

**(REGISTRO)**



### **INTERFAZ DE USUARIO**

**(INICIO DE SESIÓN)**

****

### **CATÁLOGOS DE PRODUCTOS**



### **GEOLOCALIZACIÓN**



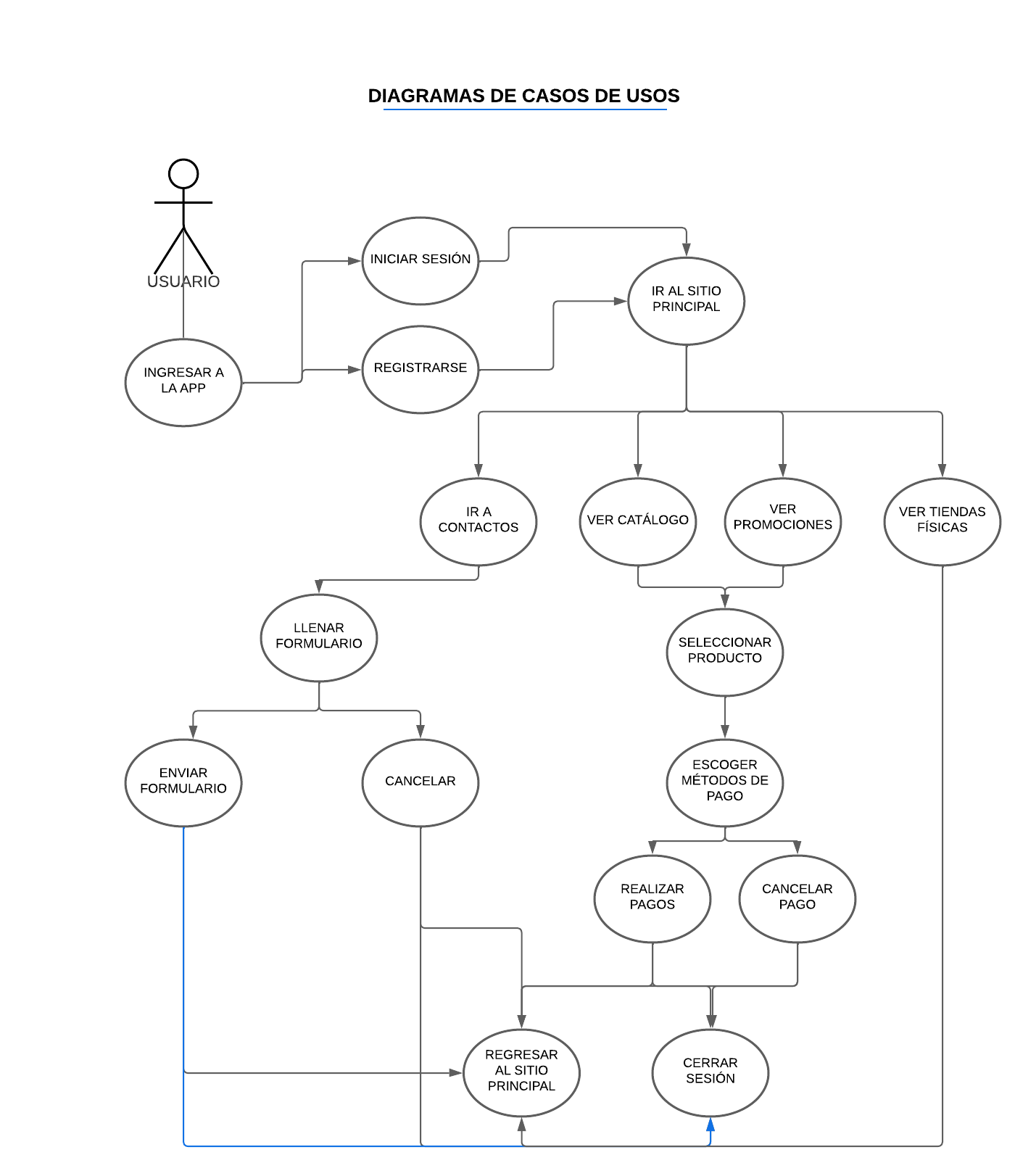
### **PROCESO DE PAGO**



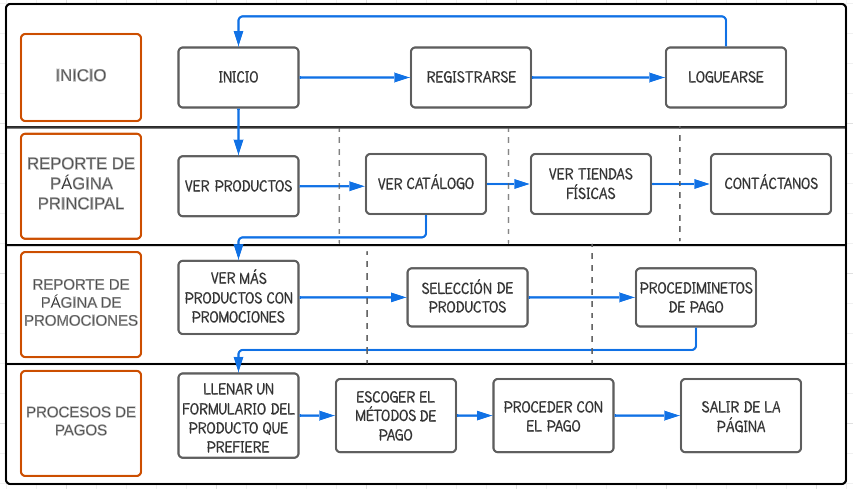
### **BARRA DE NAVEGACIÓN**



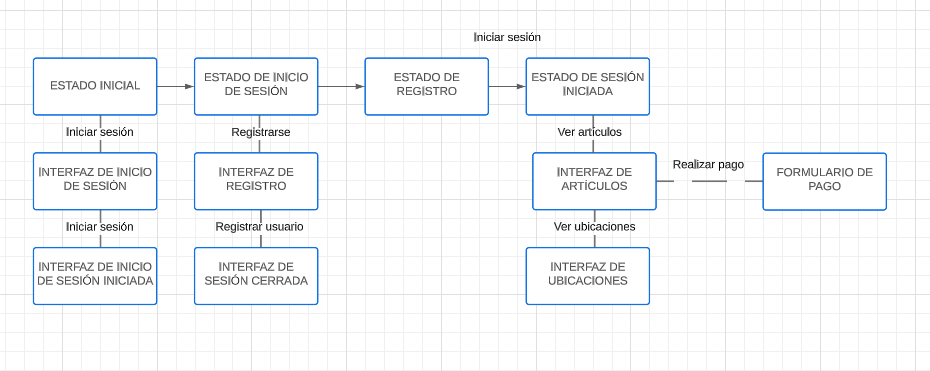
# **DIAGRAMA DE CASOS DE USOS**



# **DIAGRAMA DE CASOS DE SECUENCIA**



# **DIAGRAMA DE ESTADO**



**DICCIONARIO DE DATOS**

**LOGIN:**

**1:Importaciones:**

firebase\_auth/firebase\_auth.dart: Importa el paquete firebase\_auth, que proporciona funcionalidad para la autenticación de Firebase.

flutter/material.dart: Importa el paquete material de Flutter, que contiene widgets y utilidades para la interfaz de usuario.

**2:Clase LoginPage:**

-Extiende de StatefulWidget, lo que indica que este widget tiene un estado mutable.

-Sobrescribe el método createState() para crear y devolver una instancia de \_LoginState.

**3:Clase \_LoginState:**

-Extiende de State<LoginPage>, lo que indica que este widget tiene un estado que depende de la clase LoginPage.

-Define variables utilizadas en la pantalla de inicio de sesión.

\*\_obscureText: Un booleano que indica si la contraseña debe ocultarse o mostrarse.

\*email: Almacena el valor del campo de entrada de correo electrónico.

\*password: Almacena el valor del campo de entrada de contraseña.

\*error: Una cadena que cambia según el tipo de error encontrado.

\*\_formKey: Una clave global que identifica el formulario y se utiliza para realizar validaciones y guardar los valores del formulario.

-Override del método initState(): Es un método de ciclo de vida que se ejecuta cuando el estado del widget es inicializado. No hace nada en este caso.

-Override del método build(): Construye y devuelve la representación visual del widget.

\*Utiliza un Scaffold como el widget principal para proporcionar una estructura básica de la pantalla.

\*El AppBar muestra el título de la aplicación.

\*El body es un Container que contiene la imagen de fondo y el contenido principal.

\*Dentro del Container, se utiliza un SingleChildScrollView para permitir desplazamiento si el contenido excede el tamaño de la pantalla.

\*El contenido principal consiste en una columna vertical (Column) que contiene varios widgets:

-Una imagen de la empresa.

-Un texto que muestra el título de inicio de sesión.

-Un posible mensaje de error.

-Un formulario que contiene campos de entrada de correo electrónico y contraseña.

-Un botón para iniciar sesión.

-Un widget para registrarse.

**\*Método formulario():**

-Construye el formulario que contiene los campos de entrada de correo electrónico y contraseña.

-Utiliza un Form como widget principal y se le asigna la clave \_formKey.

-Dentro del Form, se coloca una columna vertical (Column) que contiene los campos de entrada.

**\*Método buildEmail():**

-Construye el campo de entrada de correo electrónico.

-Utiliza un TextFormField que muestra un campo de entrada de texto.

-Se le proporciona un decorador para el estilo del campo, incluyendo una etiqueta, un borde y un color de relleno.

-Se especifica el tipo de teclado (keyboardType) como TextInputType.emailAddress para mostrar el teclado adecuado.

-Se realiza validación y se guarda el valor del campo en la variable email.

**\*Método buildPassword():**

-Construye el campo de entrada de contraseña.

-Utiliza un TextFormField similar al campo de entrada de correo electrónico, pero con la opción obscureText para ocultar la contraseña.

-Se proporciona un botón de icono (IconButton) para alternar la visibilidad de la contraseña oculta o mostrada.

-Se realiza validación y se guarda el valor del campo en la variable password.

**\*Método buttonLogin():**

-Construye el botón de inicio de sesión.

-Utiliza un ElevatedButton con un estilo elevado.

-Al hacer clic en el botón, se llama a la función login() para realizar la validación de la contraseña y obtener las credenciales de usuario.

**\*Método Registrarse():**

-Construye el widget de registro.

-Utiliza un FractionallySizedBox para ajustar el tamaño del botón de registro en relación con un factor de tamaño fraccional.

-Utiliza un TextButton para mostrar un botón de registro simple.

Método login(String email, String password):

-Realiza la validación de la contraseña y devuelve las credenciales de usuario si son válidas.

-Utiliza la clase FirebaseAuth para autenticar al usuario con el correo electrónico y la contraseña proporcionados.

-Si la autenticación es exitosa, devuelve las credenciales del usuario.

-Si hay un error, muestra un mensaje de error correspondiente en la variable error.

**SIGN IN:**

**Clase UserPage:**

- Extiende: StatefulWidget

**- Importa:**

- 'package:flutter/material.dart'

- 'package:firebase\_auth/firebase\_auth.dart'

- 'package:navarrete/screens/HomePage.dart'

**- Atributos:**

- email: String (almacena el valor del campo de entrada de correo electrónico)

- password: String (almacena el valor del campo de entrada de contraseña)

- \_formKey: GlobalKey<FormState> (clave global para identificar y validar el formulario)

- error: String (almacena el mensaje de error en caso de que ocurra uno)

**- Métodos:**

- createState(): State (crea y devuelve una instancia de \_UserPageState)

- initState(): void (método del ciclo de vida que se ejecuta al inicializar el estado del widget)

- build(BuildContext context): Widget (construye la interfaz de usuario del widget)

- formulario(): Widget (devuelve el widget del formulario de registro)

- buildEmail(): Widget (devuelve el widget del campo de texto para el correo electrónico)

- buildPassword(): Widget (devuelve el widget del campo de texto para la contraseña)

- buttonCrearUsuario(): Widget (devuelve el widget del botón "Registrarse")

- crear(String email, String password): Future<UserCredential?> (crea un nuevo usuario utilizando Firebase Authentication)

**Clase \_UserPageState:**

- Extiende: State<UserPage>

- Atributos:

- Ninguno

- Métodos:

- initState(): void (método del ciclo de vida que se ejecuta al inicializar el estado del widget)

- build(BuildContext context): Widget (construye la interfaz de usuario del widget)

- formulario(): Widget (devuelve el widget del formulario de registro)

- buildEmail(): Widget (devuelve el widget del campo de texto para el correo electrónico)

- buildPassword(): Widget (devuelve el widget del campo de texto para la contraseña)

- buttonCrearUsuario(): Widget (devuelve el widget del botón "Registrarse")

- crear(String email, String password): Future<UserCredential?> (crea un nuevo usuario utilizando Firebase Authentication).

**PÁGINA PRINCIPAL:**

* **Clase ‘Paginaprincipal’ :**
  + **Atributos:**
    - **‘\_paginaActual’**: (Tipo: Int ) Índice de la página actual en la barra de navegación inferior.
    - **‘images’**: (Tipo: String) Lista de cadenas de texto que representan las URL de las imágenes utilizadas en el primer carrusel.
    - **‘imagesmarca’**: (Tipo: String) Lista de cadenas de texto que representan las URL de las imágenes utilizadas en el segundo carrusel.
  + **Métodos:**
    - **‘build(BuildContext context)’**: Método encargado de construir y devolver la interfaz de usuario de la página inicial.
    - **‘\_swiper()’**: Método privado que devuelve un widget de carrusel que muestra las imágenes de la lista images.
    - **‘\_swiper2()’**: Método privado que devuelve un widget de carrusel que muestra las imágenes de la lista **‘imagesmarca’**.
* Swiper: Clase que representa un widget de carrusel deslizante.
  + **Métodos:**
    - **‘new SwiperPagination()’** : Agrega la paginación al carrusel para permitir la navegación entre los elementos.
    - **‘new SwiperControl()’**:Agrega controles de navegación al carrusel para permitir la desplazarse hacia la izquierda y derecha.

**OFERTAS:**

**- Clase `**MyOferta**`:**

**- Atributos:**

**·** `images`: (Tipo: `String`) Imagen del logo.

· `imagesmarcas`: (Tipo: ` String `) Imágenes de la Marca.

· `paginaActual`: (Tipo: ` String `) Imagen de la página Actual.

**- Lista `Ofertas`:**

**- Tipo:** `StatelessWidget`

**- Descripción:** Es un tipo de widget en Flutter que no tiene estado mutable

**·** Clase `MyOfertas`:

· Herencia: `StatelessWidget`

**- Atributos:**

**·**  `build`: Que devuelve una instancia de.

· `MaterialApp`: Configurando las rutas y la página inicial de la aplicación.

**- Métodos:**

**·** ` main()` : Es el punto de entrada de la aplicación Flutter. Llama al constructor RunApp con una instancia de MyOferta.

· ` build(BuildContext context) `: Es el método que construye la interfaz de usuario de la aplicación en función de los datos proporcionados. Devuelve un widget de la clase.

· `createState() `: Es un método de la clase Ofertas que devuelve una nueva instancia del estado mutable asociado a este widget.

· ` build(BuildContext context)` : Es el método que construye la interfaz de usuario de la página de ofertas en función del estado actualizado. Devuelve un widget de la clase Scaffold que contiene una barra de aplicación, un cuerpo de lista y otros widgets anidados que representan las ofertas.

· `swiper(): Es un método privado de la clase \_OfertasState que devuelve un widget de la clase Swiper para mostrar las imágenes de las ofertas.

**UBICACIONES:**

**- Clase `Location`:**

**- Atributos:**

· `**name**`: (Tipo: `**String**`) Nombre de la ubicación.

· **`latitude**`: (Tipo: `**double**`) Latitud de la ubicación.

· **`longitude`:** (Tipo: **`double`)** Longitud de la ubicación.

· **`address`:** (Tipo: **`String`)** Dirección de la ubicación.

· **`contact`:** (Tipo: **`String`)** Información de contacto de la ubicación.

- Lista **`locations`:**

- Tipo: **`List<Location>`**

- Descripción: Una lista de objetos de tipo `**Location`** que contiene información de varias ubicaciones.

· Clase `**MyUbicacion`**:

· Herencia: **`StatefulWidget**`

- Atributos:

· **`\_paginaActual`:** (Tipo: `**int**`) Índice de la página actual en la barra de navegación inferior.

· **`\_tabController`:** (Tipo: **`TabController`**) Controlador de pestañas para las ubicaciones.

**- Métodos:**

· **`initState () `:** Inicializa el estado del widget y el controlador de pestañas.

· **`build (BuildContext context) `:** Construye y devuelve el árbol de widgets para el widget actual.

· **`buildMapPage (Location location) `:** Construye y devuelve el árbol de widgets para la página de mapa de una ubicación específica.

· **`openMaps (double latitude, double longitude) `:** Abre la aplicación de Google Maps con las coordenadas de latitud y longitud especificadas.

**CONTACTANOS:**

**1. Variables de controladores de texto:**

**-** `nombrecontroller`: Controlador de texto para el campo de nombre.

- `apellidoscontroller`: Controlador de texto para el campo de apellidos.

- `correocontroller`: Controlador de texto para el campo de correo electrónico.

- `telefonocontroller`: Controlador de texto para el campo de teléfono.

- `asuntocontroller`: Controlador de texto para el campo de asunto.

**-** `mensajecontroller`: Controlador de texto para el campo de mensaje.

**2. Variables de estado:**

**- `**formKey`: Clave global para el formulario.

**3. Widgets utilizados:**

**- `**Scaffold`: Proporciona una estructura básica para la pantalla, como el fondo y el app bar.

**-** `AppBar`: La barra de la aplicación en la parte superior de la pantalla.

- `Padding`: Proporciona un espacio de relleno alrededor de su hijo.

- `Form`: Un widget de formulario que mantiene el estado y valida los campos de entrada.

- `SingleChildScrollView`: Un widget que permite desplazar su contenido si no cabe en la pantalla.

- `Column`: Organiza sus hijos en una columna vertical.

- `TextFormField`: Un campo de texto con validación y controlador de texto asociado.

- `MaterialButton`: Un botón que responde a las interacciones del usuario.

- `Text`: Muestra texto en la pantalla.

# **CONCLUSIONES Y RECOMENDACIONES**

A raíz de los datos relevados en la encuesta, podemos concluir que las redes sociales son capaces de ofrecer muchas herramientas y beneficios

a quienes las utilizan, pero esto solo es así cuando los usuarios las aprovechan correctamente y las utilizan con mesura y a conciencia.

Se ha logrado reflejar una característica común en todos los resultados del presente proyecto, y se expresa relacionándolo con otras investigaciones y algunos antecedentes, de tal manera que se pueda ofrecer como descubrimiento central del estudio realizado, que justifica nuestro proyecto.

Tomando en cuenta la metodología utilizada para llevar a cabo la investigación, se logra simplificar el hecho de que se han recabado favorablemente los datos requeridos que permitieron ejecutar las pautas establecidas, que hicieron que la situación presentada, mejorará considerablemente, expresando el resultado satisfactorio de la hipótesis planteada en el tema haciendo aún más simple el hecho de ilustrar los resultados.