Práctica Final – Aplicación de Gestión de Contactos con Autenticación y Chat entre Usuarios

La presente práctica final tiene como objetivo integrar de manera coherente y completa los conocimientos adquiridos a lo largo del curso. Se deberá implementar una aplicación funcional basada en **Vue.js**, complementada con los servicios de **Firebase Authentication** y **Cloud Firestore**, incorporando interacción en tiempo real, gestión segura de datos y un sistema de comunicación básico entre usuarios.

La aplicación propuesta expande el gestor de contactos, añadiendo autenticación de usuarios, persistencia en la nube, detección de contactos registrados dentro de la plataforma y la posibilidad de establecer conversaciones individuales. Este ejercicio pretende simular un entorno de desarrollo real, promoviendo la correcta estructuración del proyecto, la aplicación de buenas prácticas y la integración efectiva entre frontend y backend serverless.

1. Descripción General de la Aplicación

La aplicación consistirá en un sistema de **gestión de contactos personales** en el que cada usuario podrá almacenar, editar y consultar sus propios contactos. Para acceder a la aplicación será imprescindible autenticarse mediante correo electrónico y contraseña. Una vez iniciado sesión, el usuario podrá navegar por las distintas vistas protegidas y gestionar su información de forma segura.
El sistema incorporará una funcionalidad adicional de detección de contactos que también se encuentren registrados en la aplicación. Esta verificación se realizará mediante la comparación del correo electrónico almacenado en cada contacto con los registros existentes en Firestore. Si ambos coinciden, se habilitará la posibilidad de iniciar una conversación directa con dicho usuario a través de un **chat en tiempo real**.

En caso contrario, la aplicación ofrecerá un mecanismo de invitación para promover la incorporación del contacto a la plataforma.

2. Requisitos Funcionales

Autenticación de Usuarios

El sistema deberá incluir un módulo de autenticación implementado mediante Firebase Authentication. Se exigirá:

- Registro de nuevos usuarios empleando correo electrónico y contraseña.
- Iniciar sesión y cerrar sesión desde la interfaz de la aplicación.
- Persistencia automática del estado de autenticación tras la recarga del sitio.
- Restricción de acceso a todas las vistas privadas mediante guardas de navegación.

Al completarse el registro, deberá crearse un documento en la colección `users` de Firestore con información básica del usuario con el formato:

```
/users/{uid}
{
  uid,
  email,
  displayName,
  createdAt
}
```

Este documento permitirá identificar si un contacto forma parte de la aplicación.

Gestión de Contactos

Se reutilizarán las vistas y funcionalidades básicas desarrolladas en prácticas anteriores, ampliándolas para conectarse directamente con Firestore. Cada usuario podrá realizar operaciones **CRUD** sobre sus propios contactos, almacenándolos bajo la colección:

`/contactos/{id}`

Cada contacto deberá incluir, al menos, nombre, email, teléfono, estado (activo o inactivo) y un indicador de favorito. Cada contacto debe almacenarse así:

```
/contactos/{id}
{
  userId: uidDelUsuarioActual,
  nombre,
  email,
  telefono,
  favorito,
  activo
}
```

La aplicación debe garantizar que cada usuario **únicamente** pueda acceder a sus propios contactos.

Identificación de Contactos Registrados

Al acceder a la vista de detalle de un contacto, la aplicación debe realizar una consulta en Firestore para determinar si el correo electrónico del contacto coincide con el de algún usuario registrado. Este proceso permitirá determinar si dicho contacto forma parte de la aplicación.

Ejemplo de consulta:

```
users.where("email", "==", contacto.email)
```

- Si el contacto **está registrado**, se mostrará un botón que habilite el acceso al chat correspondiente.
- Si el contacto **no está registrado**, se informará al usuario y se ofrecerá un mecanismo de invitación (por ejemplo, la apertura de un correo preconfigurado).

Sistema de Chat en Tiempo Real

La aplicación incorporará un sistema de mensajería individual entre usuarios registrados, basado en Firestore. Cada conversación será identificada mediante un `chatId` generado a partir de los UID de ambos usuarios, garantizando una referencia única.

Los mensajes deberán almacenarse en la estructura:

```
/chats/{chatId}/messages/{messageId}
```

El `chatId` debe generarse ordenando los UID:

```
chatId = sort([uidA, uidB]).join('_')
```

Un mensaje contiene:

```
{
  from: uidActual,
  to: uidDestino,
  text,
  timestamp,
  read
}
```

Cada mensaje incluirá información sobre el emisor, el destinatario, el contenido textual, la marca temporal y el estado de lectura. La interfaz deberá actualizarse en tiempo real usando onSnapshot, mostrando la conversación de forma clara y diferenciando visualmente los mensajes enviados de los recibidos.

La vista del chat debe:

- Mostrar mensajes en orden cronológico.
- Mostrar mensajes alineados a izquierda/derecha.
- Actualizarse en tiempo real con `onSnapshot`.
- Incluir campo de texto y botón **Enviar**.

Invitación a Usuarios No Registrados

Cuando un contacto no pertenezca a la plataforma, la aplicación deberá permitir que el usuario lo invite de forma sencilla. Se aceptarán soluciones como la generación de un correo electrónico mediante `mailto:` o la visualización de un enlace directo hacia el formulario de registro.

3. Requisitos Técnicos

La aplicación deberá desarrollarse íntegramente con **Vue 3**, utilizando la **Composition API** y organizando el código de forma modular.
Se exigirá el uso de:

- **Pinia** para la gestión del estado global. (Si se estima necesario).
- **Vue Router** para el manejo de rutas, incluyendo rutas protegidas.
- **Firebase Authentication** para la gestión segura de sesiones.
- **Cloud Firestore** como base de datos en tiempo real.
- Buenas prácticas de interfaz y correcta organización del proyecto (carpetas, stores, servicios y componentes).

4. Seguridad y Reglas de Firestore

El estudiantado deberá redactar reglas de seguridad adecuadas para Firestore, garantizando:

- El acceso sólo por parte de usuarios autenticados.
- La privacidad de los contactos, impidiendo el acceso a contactos ajenos.
- La correcta protección de la colección `users`, permitiendo la modificación únicamente por parte del usuario propietario.
- El acceso a chats limitado exclusivamente a los dos usuarios participantes en cada conversación.

Las reglas deberán entregarse junto con la memoria explicativa, justificando las decisiones tomadas.

5. Aspectos relevantes

Dado que la aplicación se desarrollará utilizando **PrimeVue** como librería de componentes, es importante **no abusar de estilos CSS personalizados**. Siempre que sea posible, se deben aprovechar las clases, temas y props de estilo que ya ofrece la librería, limitando el uso de CSS propio a pequeños ajustes puntuales (márgenes, alineaciones específicas, etc.). El objetivo es mantener una interfaz homogénea, coherente con el diseño de PrimeVue, y evitar sobrecargar el código con hojas de estilo extensas y difíciles de mantener.

6. Criterios de Evaluación

Criterio	%
Autenticación con Firebase	10%
Gestión de contactos en Firestore	20%
Comprobación de contactos registrados	15%
Chat 1:1 en tiempo real	25%
UX, estado global, rutas protegidas	10%
Creatividad (Ver anexo)	10%
Memoria*, claridad y calidad del código	10%

*La memoria es un requisito, sin la memoria la práctica no se evalúa

7. Entrega

Para la entrega se valorarán especialmente la claridad del código y la correcta estructuración de las vistas de la aplicación. Además, se deberá realizar un informe en PDF que explique, con capturas de pantalla, cada una de las funcionalidades implementadas y los aspectos más relevantes de la solución. Sin este informe, la práctica no será evaluable.

Cuando termines, comprime el proyecto completo, elimina la carpeta node_modules y súbelo con el nombre: **GestorContactos_NombreApellido1Apellido2**.zip

Anexo

1. Creatividad

Se valorará la creatividad aplicada en el desarrollo de la aplicación, entendida como la capacidad del estudiante para ampliar, enriquecer o mejorar la funcionalidad básica propuesta en el enunciado. Este criterio pretende reconocer aquellas aportaciones que, sin ser estrictamente obligatorias, demuestren iniciativa, atención a la calidad del producto final y un enfoque personal tanto en el diseño como en la experiencia de usuario.

Entre las mejoras creativas que pueden considerarse (una, varias o cualquier otra propuesta totalmente diferente), destacan (sin limitarse exclusivamente a ellas) las siguientes:

- Incorporación de **imágenes de perfil**, tanto para el usuario autenticado como para los contactos registrados en la aplicación, almacenándolas mediante Firebase Storage u otro servicio equivalente.
- **Visualización enriquecida de usuarios**, mostrando fotografía, nombre, correo electrónico u otros datos relevantes en listados, tarjetas o en la interfaz del chat.
- Mejoras significativas en la **interfaz de usuario**, incluyendo diseño responsivo avanzado, animaciones, transiciones o una estética especialmente cuidada.
- **Ampliación funcional del chat**, integrando indicadores de escritura (“typing”), envío de emojis, estados en línea u otras mejoras relacionadas con la comunicación.
- Cualquier otro añadido original que respete los principios de **usabilidad, coherencia y seguridad** del sistema.

La creatividad no se limita al plano estético: también se valorará cualquier innovación en la **arquitectura**, modularidad o diseño técnico que aporte claridad, mantenibilidad o nuevas capacidades a la aplicación.