Lector Global

Lector Global es una aplicación educativa multiplataforma (Android, Web y Windows) desarrollada en Flutter. Está diseñada para fortalecer la comprensión lectora en todos los idiomas y niveles, con una interfaz accesible, validación por correo electrónico y flujos profesionales de usuario. Inspirada en los principios del Marco Común Europeo de Referencia para las Lenguas, esta app es ideal para usuarios, escuelas, y proyectos de aprendizaje autónomo.

Filosofía del proyecto

- **Lectura para transformar**: la comprensión lectora es la puerta de entrada al pensamiento crítico y a la transformación social.
- **Global desde el inicio**: desde la primera pantalla, Lector Global es multilingüe, accesible e inclusivo.
- **Seguridad real**: autenticación sólida, validación de correo y protección contra cuentas falsas.
- **Arquitectura profesional**: modular, escalable y mantenible por cualquier equipo de desarrollo.
- **Interoperabilidad**: compatible con Android, Web y Windows desde una misma base de código.

Instrucciones de instalación

Clona el repositorio y ejecuta el proyecto en tu plataforma preferida:

```
bash git clone https://github.com/giovannihr2004/lg2.git cd lg2 flutter pub get flutter run -d chrome # Para Web flutter run -d windows # Para Windows flutter run -d <device_id> # Para Android físico o emulador
```

> Asegúrate de tener configurado tu entorno con `flutter doctor` antes de iniciar.

Plataformas soportadas

Lector Global ha sido probado y optimizado para:

- Android (APK firmado y compatible)
- Web (responsive y funcional en Chrome, Firefox)
- Windows (ejecutable `.exe` con Flutter)

En todas las plataformas se garantiza:

- Navegación suave
- Idiomas sincronizados
- Funcionalidades equivalentes

Funcionalidades actuales

| Categoría | Detalles |
|---|---| | Autenticación | Correo y contraseña, Google Sign-In (web y Android), verificación por email obligatoria | | Internacionalización| Más de 9 idiomas listos: Español, Inglés, Francés, Italiano, Portugués, Alemán, Ruso, Japonés, Chino, Árabe | | Validaciones | Checklist visual para contraseñas, validación de campos en tiempo real | | Flujo inteligente | Splash → idioma → registro/login → verificación → dashboard | | Recuperación de contraseña | Integrado con Firebase y validación directa en campo de email | | ♿ Accesibilidad | Navegación por teclado, contraste alto, compatibilidad con lectores | | Estado Global | Provider para tema, idioma y usuario | | Modularidad | Carpetas por pantallas, servicios, proveedores, widgets, l10n, etc. |

Estructura del proyecto

El proyecto está organizado con una arquitectura limpia, escalable y mantenible, lo cual permite una colaboración efectiva y facilita la evolución del sistema a futuro.

```
text lib/ ├── main.dart # Punto de entrada principal de la app │ ├── screens/ # Pantallas agrupadas por flujo lógico │ ├── auth/ # Registro, login, recuperación, verificación │ ├── splash/ # Pantallas de presentación y transiciones │ ├── intro/ # Selector de idioma y bienvenida │ └── dashboard/ # Pantalla principal tras inicio de sesión │ ├── services/ # Lógica desacoplada de autenticación y APIs │ ├── firebase_auth_service.dart # Autenticación por correo y recuperación ┬ └── google_sign_in_service.dart # Login con Google para web y Android │ ├── providers/ # Control de estado global (Provider) │ ├── language_provider.dart # Cambios dinámicos de idioma │ └── auth_provider.dart # Manejo de sesión de usuario │ ├── widgets/ # Componentes visuales reutilizables │ ├── language_selector.dart │ └── custom_button.dart │ ├── l10n/ # Archivos `.arb` para internacionalización │ ├── app_en.arb │ ├── app_es.arb │ └── (otros idiomas) │ ├── assets/ # Recursos estáticos │ ├── images/ # Íconos, ilustraciones y logotipo │ │ └── logo2.png # Ícono de aplicación personalizado │ └── lang/ # Recursos adicionales multilingües │ └── generated/ # Archivos generados automáticamente
```

Flujo de navegación del usuario

El flujo de usuario está estructurado para una experiencia lógica, fluida y segura:

1. `SplashScreen` con animaciones y logotipo
2. `LanguageSelectorScreen` (pantalla multilingüe)
3. `LoginScreen` o `RegisterScreen` (según la elección del usuario)
4. `EmailVerificationScreen` (si el usuario no ha validado su correo)
5. `DashboardScreen` (pantalla principal tras autenticación)

Todo el flujo está controlado por `SplashWrapperScreen`, que valida automáticamente si el usuario está autenticado y si su correo ha sido verificado antes de continuar.

UI/UX y diseño visual

El diseño de la app está pensado para ser funcional, accesible y elegante:

- Colores principales: `Colors.deepPurple[50]` (fondo) + `deepPurple` y `amberAccent` para elementos interactivos.
- Tipografías jerarquizadas: títulos con `FontWeight.w900`, textos con `w500`.
- Responsive: uso de `MediaQuery`, `Wrap`, `LayoutBuilder` para soportar móviles, web y escritorio.
- ♿ Accesibilidad: navegación por teclado, `Focus`, `Semantics`, y retroalimentación visual clara.
- Animaciones sutiles: transiciones suaves entre pantallas y botones resaltados al enfoque.

Autenticación y seguridad

Lector Global implementa un sistema de autenticación robusto y seguro basado en Firebase Authentication, con soporte completo para correo electrónico, contraseñas seguras y login con Google.

✉ Registro con verificación de correo

- El usuario debe ingresar un correo válido y una contraseña que cumpla con los requisitos mínimos (8 caracteres, mayúscula, minúscula, número y símbolo).
- Una vez registrado, se envía automáticamente un **correo de verificación**.
- El acceso a la aplicación queda bloqueado hasta que el correo sea confirmado.
- Este flujo garantiza que no existan cuentas falsas ni inactivas.

Inicio de sesión

- Se permite login mediante:
  - Correo y contraseña
  - Cuenta de Google (Web y Android)
- El sistema valida si el correo fue verificado antes de permitir el acceso al dashboard.

Recuperación de contraseña

- El formulario de inicio de sesión incluye un enlace **¿Olvidaste tu contraseña?**.
- Al hacer clic, el usuario es redirigido a una pantalla donde introduce su correo.
- Firebase envía automáticamente un correo con el enlace para restablecer su contraseña.

Seguridad de contraseñas

- La contraseña se valida en tiempo real durante el registro, mostrando un checklist visual con:
  - ✔ Longitud mínima
  - ✔ Letra mayúscula
  - ✔ Letra minúscula
  - ✔ Número
  - ✔ Símbolo
- El botón de registro solo se habilita cuando todos los criterios están cumplidos.

Protección de datos

- Firebase Auth maneja los datos de forma cifrada.
- No se almacenan contraseñas en texto plano.
- La validación de tokens y sesiones es gestionada por Firebase directamente.

Lógica de verificación condicional

El archivo `SplashWrapperScreen.dart` se encarga de:

- Detectar si hay un usuario autenticado (`FirebaseAuth.instance.currentUser`)
- Validar si su correo ha sido verificado (`user.emailVerified`)
- Redirigir:
  - Al dashboard si el usuario está verificado
  - A la pantalla de verificación si no lo está
  - Al selector de idioma si no hay sesión activa

Esto garantiza una **navegación automática y segura** basada en el estado real del usuario.

Configuración de Firebase

Lector Global utiliza **Firebase Authentication** como base de su sistema de usuarios. La integración está cuidadosamente preparada para funcionar en todas las plataformas soportadas.

Configuración general

1. Crea un proyecto en Firebase Console.
2. Agrega las plataformas necesarias:
    - Android
    - Web
    - Windows (con configuración opcional de claves manuales si se requiere)
3. Habilita el método de autenticación por:
    - Correo y contraseña
    - Google (asegúrate de configurar correctamente los SHA-1/SHA-256 en Android)

Archivos claves del proyecto

| Archivo | Propósito |
|---|---| | `firebase_options.dart` | Contiene los tokens y configuraciones para cada plataforma generados por `flutterfire cli` | | `firebase_auth_service.dart` | Lógica de login, registro, verificación y recuperación | | `google_sign_in_service.dart` | Integración completa con Google Sign-In | | `main.dart` | Inicialización global de Firebase y enrutamiento general|

Configuración multiplataforma

Android

- Se usa el archivo `google-services.json` ubicado en: `android/app/google-services.json`
- El plugin `com.google.gms.google-services` se encuentra registrado en `android/app/build.gradle`.

Web

- Se genera y configura el bloque de Firebase Web desde: `bash flutterfire configure`
- El archivo `index.html` contiene el `firebaseConfig`.

Windows

- Firebase Auth no tiene soporte nativo directo en escritorio, pero el flujo de autenticación (Google y correo) está preparado para ser modularizado en futuras versiones.

Comandos útiles de desarrollo

```bash

# Actualizar dependencias

flutter pub get

# Verificar que todo esté en orden

flutter doctor

# Ejecutar en Chrome (Web)

flutter run -d chrome

# Ejecutar en Android

flutter run -d

# Ejecutar en Windows

flutter run -d windows

# Generar íconos a partir de logo2.png

flutter pub run flutterlaunchericons:main

# Limpiar caché y reconstruir

flutter clean && flutter pub get ```
```

Recuerda mantener `firebase_core`, `firebase_auth` y `firebase_auth_web` en versiones compatibles entre sí para evitar conflictos de compilación.

Pruebas y control de calidad

Lector Global está siendo construido bajo buenas prácticas de desarrollo, incluyendo verificación visual, validaciones en tiempo real y soporte básico para pruebas unitarias.

Validaciones automáticas implementadas

- Validación visual de campos con íconos y bordes en tiempo real.
- Checklist de requisitos de contraseña.
- Botones desactivados hasta que el formulario esté correcto.
- Mensajes personalizados de error contextual.
- Enfoque (`Focus`) automático entre campos.

Pruebas funcionales (manuales)

| Prueba | Estado | |----------|----------| | Registro con correo válido | Funcional | | Verificación de correo | Funcional | | Login con correo no verificado | Bloqueado | | Recuperación de contraseña | Funcional | | Login con Google | Funcional | | Cambio dinámico de idioma | Funcional | | Navegación condicional según estado | Funcional |

♿ Accesibilidad integrada

Lector Global está comprometido con la inclusión digital. Las siguientes medidas han sido implementadas:

- Soporte completo para navegación por teclado (`Focus`, `FocusNode`)
- Iconos e indicadores visibles cuando el elemento está seleccionado
- Uso de `Semantics` para compatibilidad con lectores de pantalla
- Colores con suficiente contraste y tipografía legible
- Mensajes visuales y de texto para validación accesible

Documentación integrada

El proyecto incluye documentación tanto interna como externa:

- `README.md`: guía detallada del proyecto
- Comentarios en español en todos los archivos `.dart`
- Estructura clara y nombres descriptivos en clases, variables y rutas
- Archivos `.arb` sincronizados para traducción en tiempo real
- Carpetas organizadas para facilitar la comprensión por nuevos desarrolladores

---

Despliegue

Android

```
bash flutter build apk
```

El APK se genera en:

```
build/app/outputs/flutter-apk/app-release.apk
```

Este archivo puede instalarse directamente en dispositivos físicos y está preparado para firmarse y subirse a Google Play en etapas futuras.

Web

```
bash flutter build web
```

El proyecto se compila en:

```
build/web/
```

Puede ser subido fácilmente a servicios como Firebase Hosting, GitHub Pages o cualquier servidor web estático.

Windows

```
bash flutter build windows
```

Se genera un ejecutable en:

```
build/windows/runner/Release/
```

Listo para distribución en entorno educativo o local sin necesidad de instalación adicional.

Licencia

Este proyecto ha sido desarrollado con fines educativos, sociales y culturales.
Lector Global se distribuye bajo los principios de acceso libre, mejora continua y colaboración abierta.

Permisos:

- Se permite usar y modificar el código con fines educativos o no comerciales.

- Se permite compartir públicamente el proyecto siempre que se dé el debido crédito.
- No se permite su comercialización sin autorización explícita del autor.

Una futura versión puede adoptar una licencia abierta como MIT o GPL si se consolida una comunidad activa.

Autor principal

Giovanni Holguín Rojas
Medellín, Colombia
Docente, investigador y desarrollador de software educativo.
Contacto: giovannihr2004@gmail.com
GitHub – Repositorio oficial

Comunidad y contribuciones

Lector Global está abierto a la colaboración de:

- Desarrolladores Flutter
- Docentes y pedagogos
- Traductores y correctores
- Diseñadores de experiencia de usuario (UX)
- Estudiantes apasionados por la lectura y la tecnología

Si deseas contribuir:

```bash
# 1. Haz un fork del proyecto

# 2. Crea una rama con tu mejora: git checkout -b feature/tu-aporte

# 3. Sube tus cambios: git push origin feature/tu-aporte

# 4. Abre un Pull Request desde GitHub
```

Recursos útiles

- Documentación oficial de Flutter
- Firebase para Flutter
- Provider para manejo de estado
- Google Fonts
- Material Icons

Visión futura

Lector Global está en constante evolución. Las próximas etapas planeadas incluyen:

- Sistema de estadísticas de lectura por usuario
- Modo “Lector Global para Escuelas” (gestión docente y tareas)
- Evaluaciones personalizadas por nivel y progreso
- Más de 30 idiomas soportados con IA para traducción automática
- Integración con APIs abiertas de libros, cuentos y artículos
- Gamificación del aprendizaje lector con insignias y niveles

---

> “Si puedes leer, puedes comprender.
> Y si puedes comprender, puedes cambiar tu vida.
> Y si cambiamos vidas, cambiamos el mundo.”
