Diseño de Arquitectura del Sistema – Braille Express

**Fecha:** 16/04/2025

**Autor:** Ylia Jamile Ochoa Gutierrez

**Versión:** 1.0

# 1. Introducción

Este Este documento detalla la arquitectura del sistema para la aplicación móvil Braille Express, cuyo propósito es permitir a los usuarios convertir texto a Braille y gestionar dispositivos Bluetooth para su comunicación con hardware externo. Se define la estructura general del sistema, los componentes que lo conforman, sus interacciones y las tecnologías empleadas, todo ello bajo un enfoque moderno basado en el patrón de arquitectura Model-View-ViewModel (MVVM).

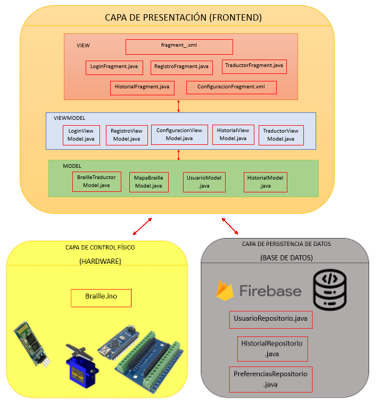
# 2. Visón General de la Arquitectura

## 2.1. Estilo Arquitectónico

El sistema se basa en una arquitectura en capas, donde cada una se corresponde con una responsabilidad claramente definida dentro del patrón MVVM. Se identifican tres capas principales:

* **Capa de Presentación**Interfaz gráfica e interacción directa con el usuario.
* **Capa de Persistencia de Datos:** Acceso y almacenamiento remoto de información (Firebase).
* **Capa de Control Físico:** Comunicación con hardware externo, como el Arduino Nano mediante Bluetooth.

## 2.2. Diagrama General de Arquitectura:



# 3. Descripción de Componentes Principales

## 3.1. Capa de Presentación

Esta capa corresponde a la interfaz gráfica con la que interactúa el usuario. Está diseñada bajo el patrón MVVM, permitiendo una separación clara entre los datos, la lógica de presentación y la vista.

* **Plataforma:** Android
* **Lenguaje:** Java
* **Modelo:** MVVM
* **Estilo visual:** XML Layouts
* **Responsabilidades:**
* Mostrar interfaces amigables.
* Capturar y validar entradas del usuario.
* Notificar errores, resultados y estados del sistema.
* Navegar entre pantallas mediante Navigation Component.
* **Views:**
* MainActivity.java: Actividad principal que contiene el menú de navegación lateral (Drawer) y gestiona los fragments.
* TraductorFragment.java/fragment\_traductor.xml: Vista para ingresar texto y visualizar su traducción en braille.
* ConfiguracionFragment.java/fragment\_configuracion.xml: Interfaz para realizar personalizaciones como temas, y color de botones.
* LoginFragment.java/fragment\_login.xml: Pantalla de inicio de sesión con validación de usuario.
* RegistroFragment.java/fragment\_registro.xml: Formulario para el registro de nuevos usuarios.
* HistorialFragment.java/ fragment\_historial.xml: Muestra las traducciones realizadas recientemente.
* fragment\_splash.xml: Diseño de presentación inicial muestra el logo y el slogan.
* fragment\_diccionario.xml: Interfaz donde muestra el alfabeto Braille por imágenes.
* **ViewModel:**
* TraductorViewModel.java: Se encarga de la conversión de texto a Braille mediante el MapaBraille.java.
* ConfiguracionViewModel.java: Encargado de gestionar la lógica relacionada con la personalización de la interfaz de usuario en la aplicación.
* LoginViewModel.java: Maneja la validación de credenciales y la sesión con Firebase.
* RegistroViewModel.java: Procesa los datos del formulario y coordina el alta de usuario.
* BluetoothViewModel.java: Coordina la comunicación con Arduino a través de Bluetooth.
* HistorialViewModel.java: Obtiene los datos desde la base de datos y los presenta al usuario.
* **Model:**
* BrailleTraductorModel.java: Modelo encargado de ejecutar la conversión de texto a braille.
* MapaBrailleModel.java: Diccionario interno que mapea caracteres a su representación braille.
* UsuarioModel.java: Clase entidad para el usuario (nombre, apellidos, DNI, etc.).
* HistorialModel.java: Clase entidad para elementos del historial.

## 3.2. Capa de Persistencia de Datos (Firebase)

* **Sistema Gestor:** Firebase
* **Componentes:**
* UsuarioRepositorio.java: Gestiona el acceso a Firebase para operaciones de creación, autenticación y recuperación de usuarios.
* HistorialRepositorio.java: Guarda y obtiene el historial de traducciones por usuario.
* PreferenciasRepositorio.java: Administra las preferencias de usuario y el estado de sesión localmente utilizando SharedPreferences.

## 3.3. Capa de Control Físico (Hardware)

* **Plataforma:** Arduino IDE
* **Responsabilidades:**
* Ejecutar el código que traduce señales recibidas vía Bluetooth en acciones físicas.
* Controlar dispositivos físicos como actuadores, servomotores o pines que representan caracteres braille en relieve.
* Interpretar los datos enviados por la app Android y convertirlos en salidas táctiles o visuales para el usuario.
* **Componentes:**
* Braille.ino: Archivo principal que contiene todo el código de control del hardware. Internamente modularizado en funciones específicas para conexión, traducción y control de servos.

# 4. Integraciones Externas (Opcionales)

# 5. Seguridad

La aplicación BrailleExpress incorpora mecanismos fundamentales para proteger tanto la integridad como la privacidad de los datos de los usuarios y el correcto funcionamiento del sistema:

* Protección de datos: Los datos del usuario (como credenciales y configuraciones) se almacenan de forma segura en la nube mediante Firebase.
* Privacidad de contenido: Por principios de privacidad, el contenido textual traducido no se almacena en servidores; únicamente se guardan configuraciones de usuario y registros mínimos del historial local.
* Comunicación Bluetooth segura: La conexión con el módulo HC-06 está restringida a dispositivos previamente emparejados, evitando accesos no autorizados. Además, se implementa un control de flujo que impide el envío de datos fuera de sesiones activas.
* Validación de entradas: Todos los campos de texto e interacciones del usuario son validados previamente, previniendo errores operativos o posibles intentos de inyección de comandos maliciosos.
* Gestión de permisos: La aplicación solicita únicamente los permisos estrictamente necesarios, en conformidad con la versión del sistema operativo, y ofrece explicaciones contextuales al usuario para mayor transparencia.

# 6. Escalabilidad y Despliegue

Aplicación móvil:

* Compatibilidad: Desarrollada para dispositivos con Android 7.0 (Nougat) o superior, asegurando compatibilidad con una amplia gama de terminales, incluso aquellos de gama media o baja.
* Despliegue flexible: Puede distribuirse en formato APK o a través de la Google Play Store, permitiendo actualizaciones automáticas y control de versiones.
* Mantenibilidad: La arquitectura MVVM garantiza una alta cohesión y bajo acoplamiento entre componentes, lo que facilita la implementación de mejoras, nuevas funcionalidades y la corrección de errores sin comprometer la estabilidad general de la aplicación.

Dispositivo físico (Arduino + Servos):

* Modularidad del hardware: El dispositivo está compuesto por componentes ampliamente disponibles (Arduino Nano, módulo Bluetooth HC-06, servos SG90 y Shield IO), lo que permite su fácil reemplazo, réplica y mantenimiento.
* Replicabilidad educativa: Esta solución puede adaptarse a diferentes contextos escolares, centros de rehabilitación o instituciones inclusivas, al estar pensada como un kit de bajo costo y sencilla implementación.

Posibles fallas técnicas:

* Alcance limitado del HC-06: El módulo Bluetooth clásico ofrece un alcance de aproximadamente 10 metros, lo que lo hace ideal para ambientes controlados o uso de escritorio.
* Limitaciones del Arduino Nano: Dada su capacidad limitada, se delega la lógica principal a la aplicación móvil, reservando al Arduino funciones simples de control físico.
* Latencia de transmisión: El envío carácter por carácter puede introducir demoras perceptibles, mitigables mediante la implementación de buffers o protocolos de codificación más eficientes.
* Compatibilidad con permisos: Android 7 permite gestionar permisos en la instalación o mediante solicitudes en tiempo real, lo que facilita la implementación de funcionalidades sin complejas dependencias del sistema.

# 7. Conclusiones

La arquitectura de Braille Express ofrece una solución robusta, modular y centrada en la accesibilidad para personas con discapacidad visual. Gracias a la adopción del patrón MVVM, se logra una clara separación de responsabilidades entre las vistas, la lógica y los datos, lo que favorece el mantenimiento, la escalabilidad y la evolución futura del sistema.

El enfoque híbrido entre software (aplicación móvil) y hardware (dispositivo Arduino) permite desarrollar una herramienta potente y accesible, funcional incluso en contextos educativos o de bajos recursos.

Entre los principales beneficios de esta solución destacan:

* Compatibilidad con una amplia base de dispositivos Android (7.0+).
* Eliminación de dependencia de servidores físicos, gracias al uso de Firebase.
* Coste reducido de implementación, al emplear componentes electrónicos económicos y de fácil acceso.
* Alta replicabilidad en entornos educativos e inclusivos, promoviendo el acceso universal a la lectura Braille.

Braille Express se presenta, así como una alternativa tecnológica inclusiva, sostenible y de alto impacto social, orientada a democratizar el acceso a la lectura y la comprensión lectora para personas con discapacidad visual, mediante el uso estratégico de herramientas móviles y hardware asequible.