Registro Diario de Avances – Sprint 1 Braille Express

**Fecha:** 13/05/2025

**Autor:** Ylia Jamile Ochoa Gutierrez

**Versión:** 1.0

# 1. Introducción

Este documento detalla el seguimiento diario del Sprint 2 del proyecto Braille Express, centrado en la implementación de funcionalidades esenciales para el módulo de inicio y cierre de sesión en la nube, así como el desarrollo inicial del ingreso manual de texto para su traducción a Braille. Se documentan las actividades diarias, tareas realizadas, fragmentos de código como evidencia del avance, un burndown chart correspondiente y un análisis del rendimiento del equipo durante las 60 horas efectivas del sprint.

# 2. Lineamientos de Diseño

Desarrollar una primera versión funcional de la aplicación TaskManager que permita a los usuarios:

* Diseño de interfaces de registro y login.
* Registro seguro de usuarios (con almacenamiento en Firebase Realtime Database).
* Validación de DNI.

# 3. Historias de Usuario y Tareas

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Backlog ID | Historia de Usuario | Tiempo Estimado | Fecha de Inicio | Fecha de Finalización |
| HU-1.2 | Inicio y cierre de sesión en la nube | 8 pts | 01/05/2025 | 09/05/2025 |
| HU-2.1 | Ingreso manual de texto | 5 pts | 09/05/2025 | 10/05/2025 |

# 4. Evidencias de Código

## 4.1 Historia de Usuario 1.2 — Inicio y cierre de sesión en la nube

Archivo: LoginFragment.java

public class LoginFragment extends Fragment {  
  
 private FragmentLoginBinding binding;  
 private LoginViewModel loginViewModel;  
 @Override  
 public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState) {  
 binding = FragmentLoginBinding.*inflate*(inflater, container, false);  
 loginViewModel = new ViewModelProvider(this).get(LoginViewModel.class);  
 // Configuración del botón de iniciar sesión  
 binding.btnLogin.setOnClickListener(view -> {  
 String dni = binding.Dni.getText().toString().trim();  
 String password = binding.password.getText().toString().trim();  
 // Verificamos si algún campo está vacío  
 if (dni.isEmpty() || password.isEmpty()) {  
 // Mostramos un mensaje de advertencia si los campos no están completos  
 Toast.*makeText*(getContext(), "Por favor, completa todos los campos", Toast.*LENGTH\_SHORT*).show();  
 return;  
 }  
 // Llamamos al metodo iniciarSesion del ViewModel para autenticar al usuario  
 loginViewModel.iniciarSesion(dni, password, isExitoso -> {  
 if (isExitoso) {  
 // Llamamos al metodo iniciarSesion del ViewModel para autenticar al usuario  
 NavHostFragment.*findNavController*(LoginFragment.this)  
 .navigate(R.id.*nav\_Traductor*);  
 } else {  
 // Si las credenciales son incorrectas, mostramos un mensaje de error  
 Toast.*makeText*(getContext(), "DNI o contraseña incorrectos", Toast.*LENGTH\_SHORT*).show();  
 }  
 });  
 });  
 // Configuración del link para ir al fragmento de registro  
 binding.linkRegistro.setOnClickListener(v -> {  
 NavHostFragment.*findNavController*(LoginFragment.this)  
 .navigate(R.id.*nav\_Registro*);  
 });  
 // Retornamos la vista raíz del fragment para que se muestre en la interfaz  
 return binding.getRoot();  
  
 }  
}

Archivo: LoginViewModel.java

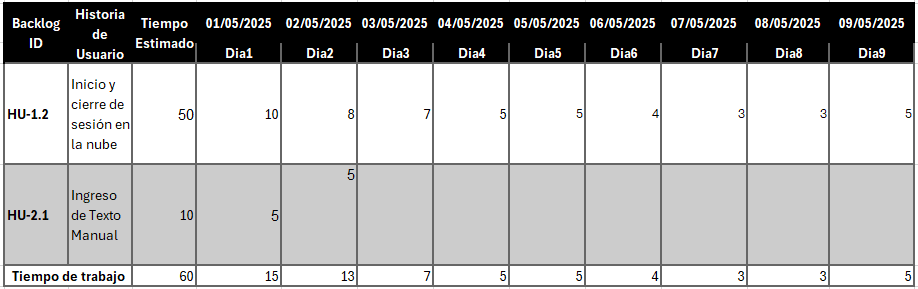
public class LoginViewModel extends ViewModel {  
  
 public interface LoginCallback {  
 void onResult(boolean isExitoso);  
 }  
 // Metodo para iniciar sesión usando el DNI y la contraseña  
 public void iniciarSesion(String dni, String password, LoginCallback callback) {  
 // Usamos Firebase para obtener los datos del usuario desde la base de datos  
 FirebaseDatabase.*getInstance*().getReference("usuarios").child(dni)  
 .addListenerForSingleValueEvent(new ValueEventListener() {  
 @Override  
 public void onDataChange(DataSnapshot snapshot) {  
 // Verificamos si el usuario existe en la base de datos  
 if (snapshot.exists()) {  
 // Obtenemos la contraseña almacenada en la base de datos para el usuario con el DNI proporcionado  
 String storedPassword = snapshot.child("Password").getValue(String.class);  
 // Si la contraseña almacenada no es nula y coincide con la contraseña ingresada  
 if (storedPassword != null && storedPassword.equals(password)) {  
 callback.onResult(true);  
 } else {  
 callback.onResult(false);  
 }  
 } else {  
 callback.onResult(false);  
 }  
 }  
 // En caso de error al intentar acceder a la base de datos  
 @Override  
 public void onCancelled(DatabaseError error) {  
 callback.onResult(false);  
 }  
 });  
 }  
}

## 4.2 Historia de Usuario 2.1 — Ingreso manual de texto

Archivo: fragment\_traductor.xml

//<!-- Estado Bluetooth + Switch -->  
 <TextView  
 android:id="@+id/txtEstadoBluetooth"  
 android:layout\_width="0dp"  
 android:layout\_height="wrap\_content"  
 android:text="Estado: Desconectado"  
 android:textColor="#FF0303"  
 android:textSize="16sp"  
 android:textStyle="bold"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"  
 app:layout\_constraintEnd\_toStartOf="@+id/swBluetooth"  
 app:layout\_constraintHorizontal\_bias="0"/>  
  
 <Switch  
 android:id="@+id/swBluetooth"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:onClick="abrirConfiguracionBluetooth"  
 app:layout\_constraintTop\_toTopOf="@id/txtEstadoBluetooth"  
 app:layout\_constraintEnd\_toEndOf="parent" />  
  
 <!-- Caja de texto de entrada -->  
 <EditText  
 android:id="@+id/txtEntrada"  
 android:layout\_width="367dp"  
 android:layout\_height="232dp"  
 android:layout\_marginTop="16dp"  
 android:background="@android:drawable/edit\_text"  
 android:gravity="start|top"  
 android:hint="Escribe el texto aquí"  
 android:inputType="textMultiLine"  
 android:padding="10dp"  
 android:scrollbars="vertical"  
 android:textSize="16sp"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toBottomOf="@id/txtEstadoBluetooth" />  
  
 <!-- Botones: Pegar - Micrófono - Copiar -->  
 <LinearLayout  
 android:id="@+id/layoutBotonesTexto"  
 android:layout\_width="0dp"  
 android:layout\_height="wrap\_content"  
 android:orientation="horizontal"  
 android:gravity="center"  
 app:layout\_constraintTop\_toBottomOf="@id/txtEntrada"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 android:layout\_marginTop="12dp">  
  
 <ImageButton  
 android:id="@+id/btnPegar"  
 android:layout\_width="70dp"  
 android:layout\_height="64dp"  
 android:background="@null"  
 android:contentDescription="Pegar texto"  
 android:src="@drawable/ic\_pegar" />  
  
 <ImageButton  
 android:id="@+id/btnHablar"  
 android:layout\_width="85dp"  
 android:layout\_height="match\_parent"  
 android:background="@null"  
 android:contentDescription="Hablar"  
 android:src="@drawable/ic\_micro" />  
  
 <ImageButton  
 android:id="@+id/btnCopiar"  
 android:layout\_width="67dp"  
 android:layout\_height="64dp"  
 android:background="@null"  
 android:contentDescription="Copiar texto"  
 android:src="@drawable/ic\_copiar" />  
 </LinearLayout>  
  
 <!-- Castellano - Traducir - Braille -->  
 <LinearLayout  
 android:id="@+id/layoutTraductores"  
 android:layout\_width="0dp"  
 android:layout\_height="wrap\_content"  
 android:orientation="horizontal"  
 android:gravity="center"  
 android:layout\_marginTop="12dp"  
 app:layout\_constraintTop\_toBottomOf="@id/layoutBotonesTexto"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent">  
  
 </LinearLayout>  
  
 <!-- Resultado en Braille -->  
 <TextView  
 android:id="@+id/txtSalida"  
 android:layout\_width="368dp"  
 android:layout\_height="234dp"  
 android:layout\_marginTop="12dp"  
 android:background="#EEEEEE"  
 android:gravity="start|top"  
 android:hint="Traducción Braille"  
 android:padding="10dp"  
 android:scrollbars="vertical"  
 android:textColor="@color/black"  
 android:textSize="16sp"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toBottomOf="@id/layoutTraductores" />  
  
 <!-- Botón Enviar -->  
 <Button  
 android:id="@+id/btnEnviar"  
 android:layout\_width="0dp"  
 android:layout\_height="wrap\_content"  
 android:text="Enviar"  
 android:backgroundTint="@color/purple\_500"  
 android:textColor="@color/white"  
 app:layout\_constraintTop\_toBottomOf="@id/txtSalida"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 android:layout\_marginTop="16dp" />  
</androidx.constraintlayout.widget.ConstraintLayout>

# 5. Burndown Chart



# 6. Análisis del Sprint

* Se logró implementar correctamente la autenticación en la nube con Firebase, incluyendo inicio y cierre de sesión desde múltiples dispositivos.
* Aunque el sprint estaba estimado en 68 horas, se completaron todas las tareas en 60 horas, lo que refleja eficiencia y una mejora en la velocidad del equipo.
* La integración con la nube permite mantener los datos del usuario disponibles y seguros, cumpliendo con los criterios de aceptación de la HU 1.2.
* Se logró diseñar la interfaz de ingreso de texto y dar los primeros pasos técnicos hacia la traducción Braille, anticipando posibles dificultades.
* Tomó tiempo establecer y validar correctamente los permisos y rutas para la autenticación, pero se resolvió satisfactoriamente.
* Al desarrollar la interfaz para usuarios con discapacidad visual, fue necesario realizar ajustes para mejorar la usabilidad y accesibilidad, lo que exigió pruebas adicionales.

# 7. Conclusiones

Durante el Sprint 2 se planificó la ejecución de seis tareas, correspondientes a la totalidad de la Historia de Usuario 1.2: Inicio y cierre de sesión en la nube y a las dos primeras tareas de la Historia de Usuario 2.1: Ingreso manual de texto. Esta planificación se realizó con el objetivo de cumplir con las 68 horas estimadas para el sprint, asegurando un avance sostenido y realista del proyecto.

La Historia de Usuario 1.2 fue completada satisfactoriamente, implementando un sistema de autenticación funcional mediante Firebase, que permite a los usuarios iniciar y cerrar sesión desde cualquier dispositivo con sincronización en la nube, garantizando así la seguridad y disponibilidad de los datos del usuario. Respecto a la Historia de Usuario 2.1, se avanzó en el diseño de la interfaz para el ingreso manual de texto.

Aunque el sprint estaba programado para 68 horas, se logró completar las tareas en 60 horas, demostrando una buena estimación del esfuerzo requerido y una ejecución eficiente del equipo de desarrollo. Este resultado también evidencia una mejora en la productividad y una mayor familiaridad con el entorno de desarrollo y las herramientas utilizadas.