Jaime Huenchulao, Alex Tardón

Practica Profecional 2 | Lab Cimm UBB

Documentación

De aplicacion

Índice

[Introducción 2](#_Toc125580945)

[Librerías, configuración y entorno de desarrollo 3](#_Toc125580946)

[Uso de la aplicación 6](#_Toc125580947)

[Código y repositorio 13](#_Toc125580948)

# Introducción

El presente documento tiene por finalidad esclarecer, detallar y especificar, las funciones de la aplicación desarrollada para ser usada en conjunto con el dispositivo de ESP32. Así como también servir como una guía rápida de uso de la aplicación.

La aplicación está diseñada para interactuar con una versión modificada de una tesis de otros alumnos que mide la deformación en paredes, el objetivo del proyecto asignado por el profesor Luis Vera, es crear una aplicación móvil capaz de interactuar con el dispositivo, facilitando su uso, permitiendo al usuario interactuar sin la necesidad de modificar el código. Además, la aplicación implementa una serie de funciones, como por ejemplo la opción de agendar el funcionamiento del dispositivo a un periodo de tiempo determinado, la opción de recibir los datos en tiempo real con un temporizador, etc.

La totalidad del código se puede encontrar en el siguiente [link](https://github.com/NikoGB/ESP32-OpticalEncoder-AndroidStudioProyect) de GitHub.

# Librerías, configuración y entorno de desarrollo

Para desarrollar el proyecto se utilizo la plataforma de desarrollo de Android Studio, la cual se puede descargar desde el siguiente [link](https://developer.android.com/studio), para la comunicación serial mediante bluetooth se utilizó la siguiente librería disponible en [GitHub](https://github.com/harry1453/android-bluetooth-serial) y para mostrar los datos de manera grafica se uso la siguiente librería, disponible también en [GitHub](https://github.com/jjoe64/GraphView). Cada librería trae sus instrucciones de instalación en sus repositorios correspondientes.

Como lenguaje de programación se utilizo java, y la aplicación se creo con las siguientes opciones o configuraciones.



Ilustración ‑ Contenidos del Build.Gradle

Debido a la necesidad de exportar los datos fuera de la aplicación y la necesidad de establecer comunicación el dispositivo mediante bluetooth, el archivo de AndroidManifest.xml contiene:

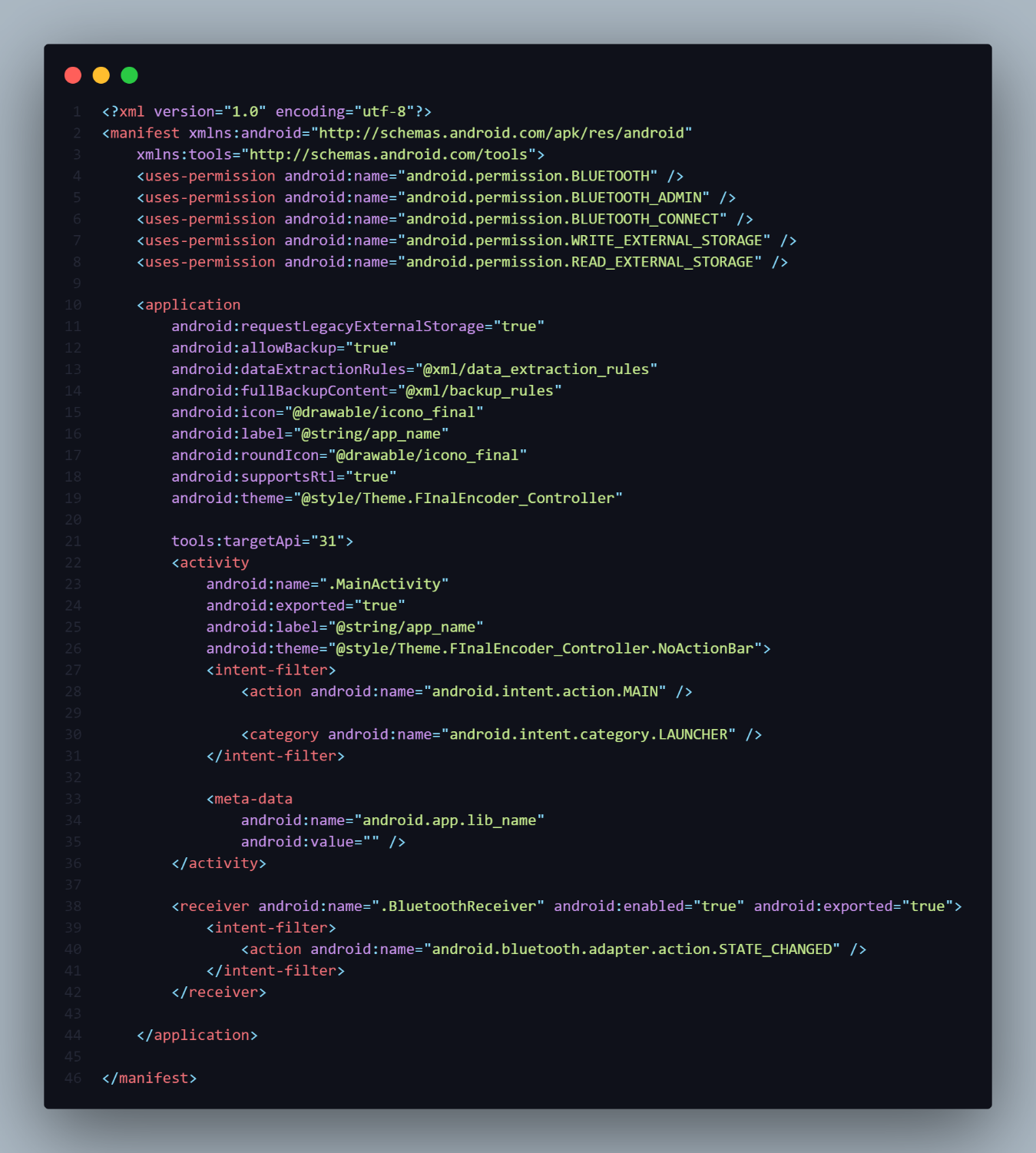


Ilustración ‑ Contenidos de AndroidManifest.xml

Los contenidos del archivo settings.gradle son los siguientes:



Ilustración ‑ Contenidos del archivo settings.gradle

Y finalmente dentro de gradle.properties se recomienda utilizar las siguientes opciones para evitar errores referentes a las librerías:

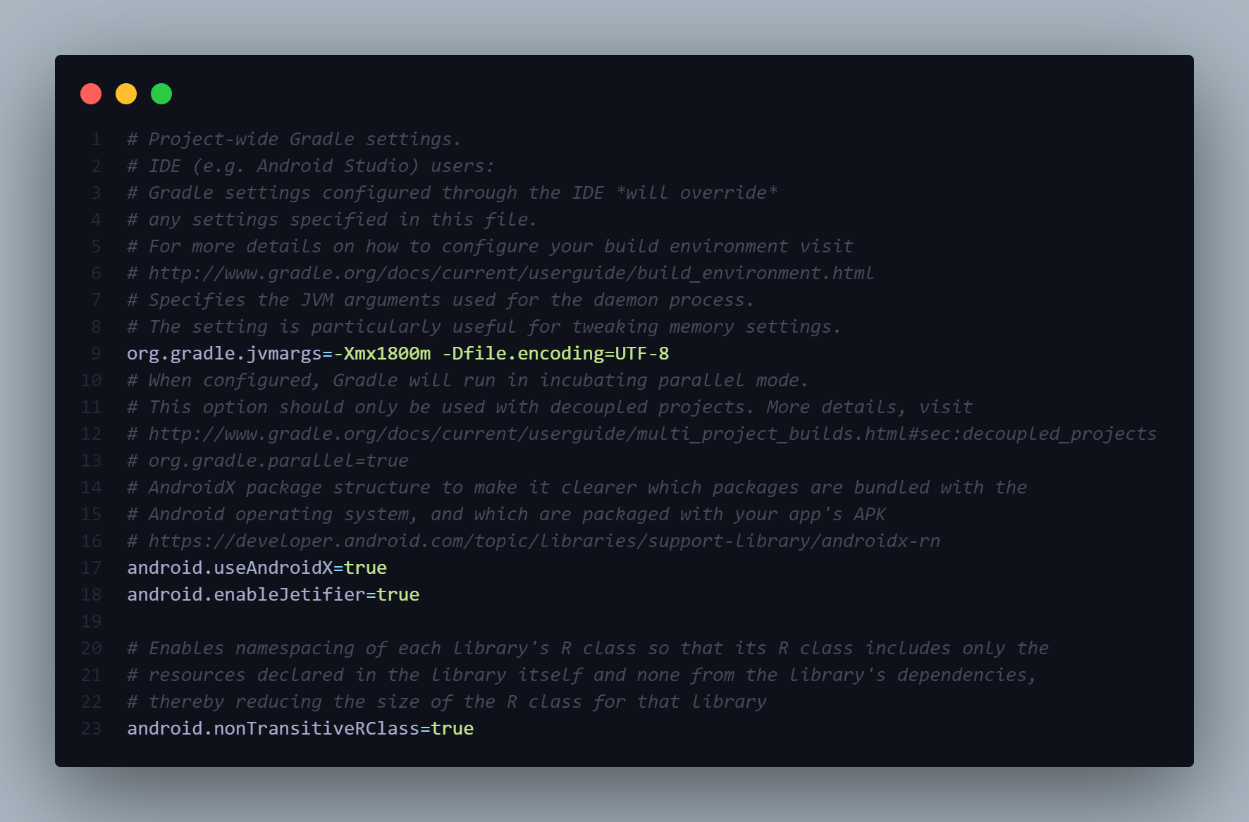


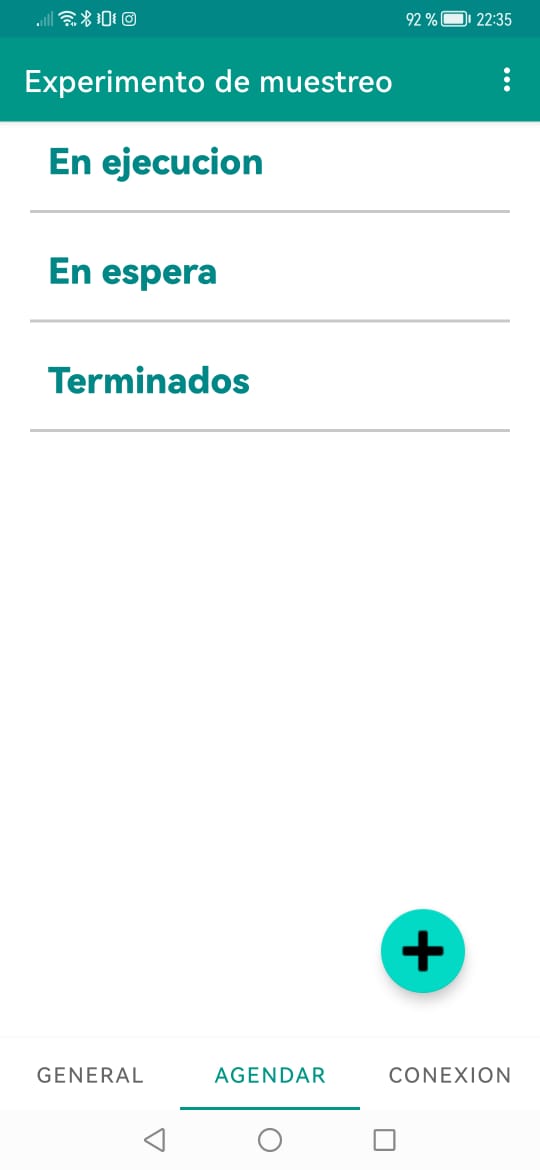
Ilustración ‑ Contenidos de gradle.properties

# Uso de la aplicación

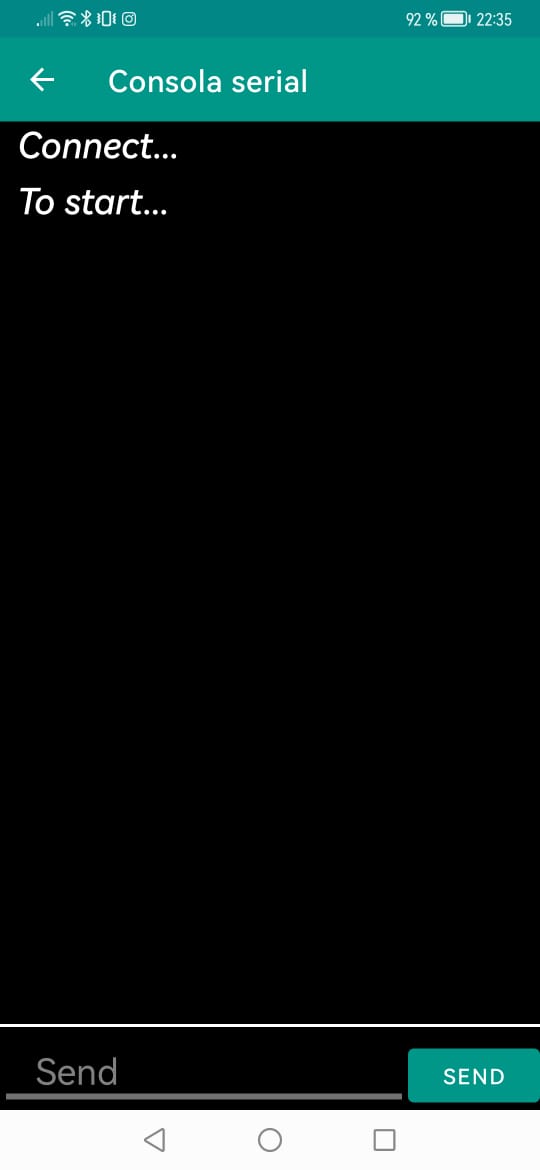
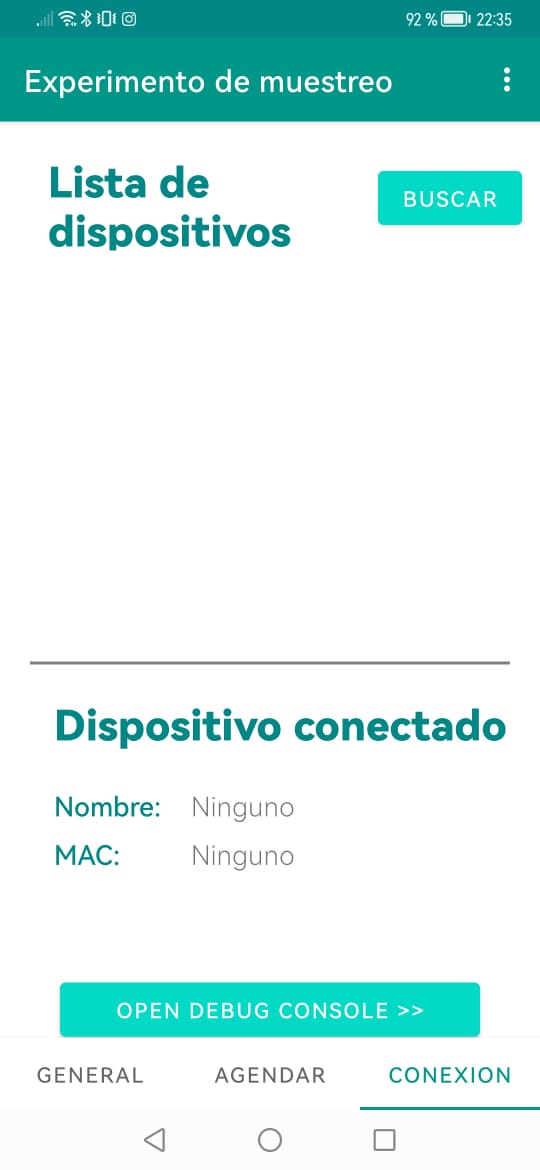
La aplicación cuenta con tres menús principales que permiten de una u otra manera interactuar con el dispositivo (esp32) al usuario, estos son “General”, “Agendar” y “Conexión”, en el menú de general se presentan las opciones de encender el muestreo (como “Estado de muestreo”), definir el Tiempo de muestreo, definir un temporizador, y un grafico para visualizar los datos recibidos.



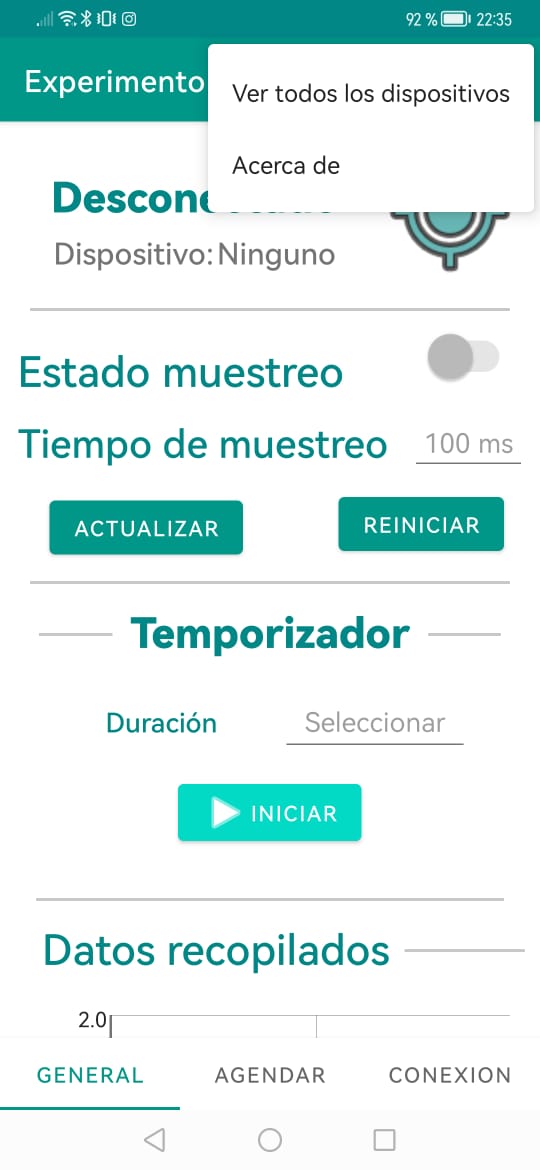
En el menú de agendar el usuario puede visualizar los distintos agendamientos que se encuentren y el estado de estos, para crear un agendamiento se debe pulsar el botón de “+” y será provisto del menú que aparece en la imagen de la derecha.



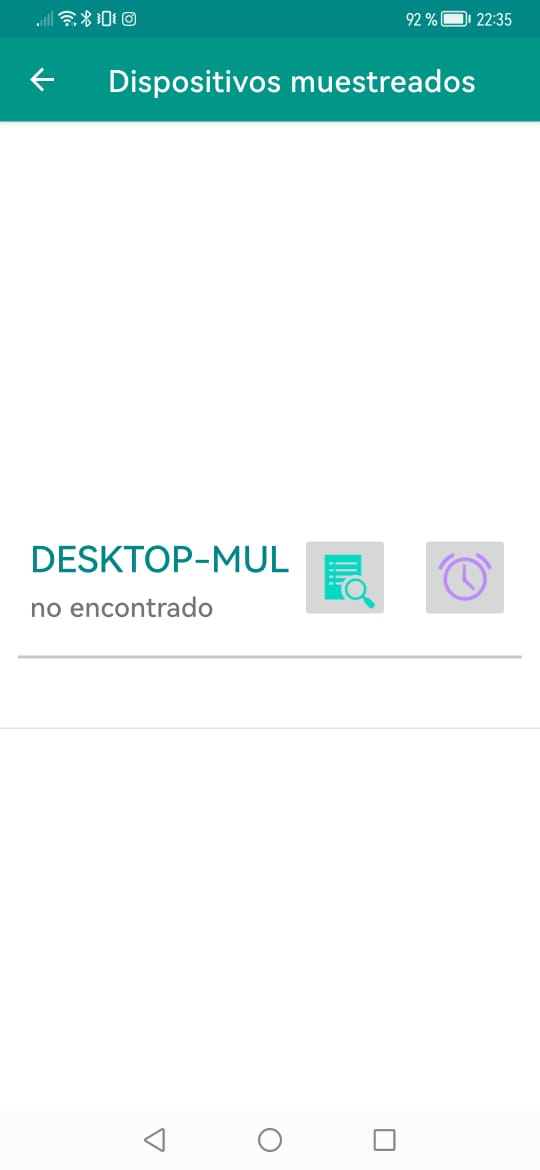
En el menú de conexión se presenta la lista de dispositivos y el botón para buscar los dispositivos enlazados, el dispositivo ESP32 DEBE estar enlazado antes de conectarse, de lo contrario no se vera en la lista de dispositivos. Además de mostrar el nombre y la dirección MAC del dispositivo, el usuario, si así lo requiriese, puede acceder a la consola de debug (imagen de la derecha), la cual le permite ver todos los mensajes recibidos por la aplicación, solo se recomienda en caso de ser necesario.



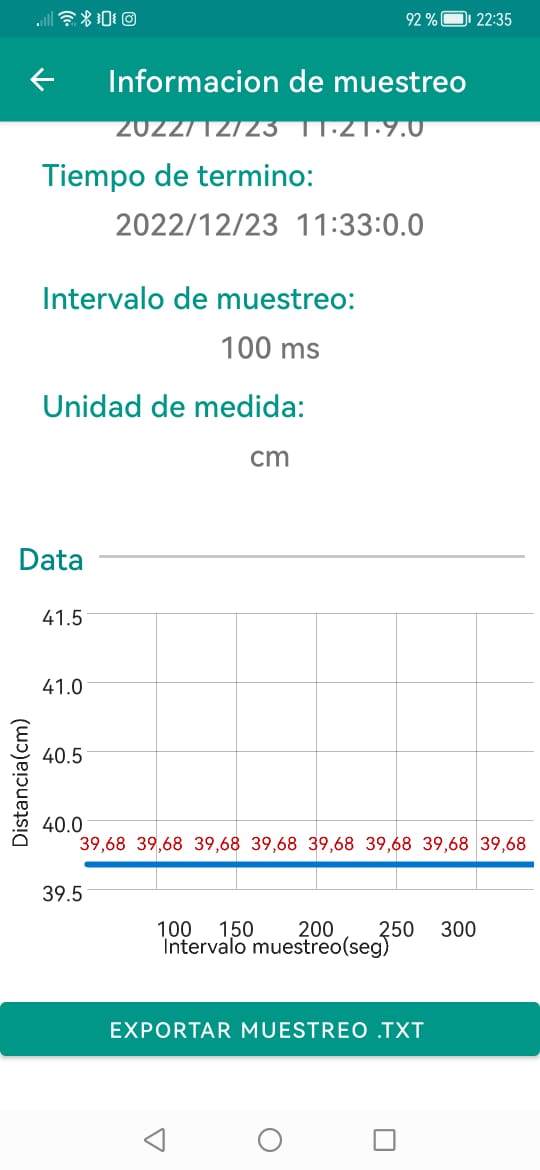
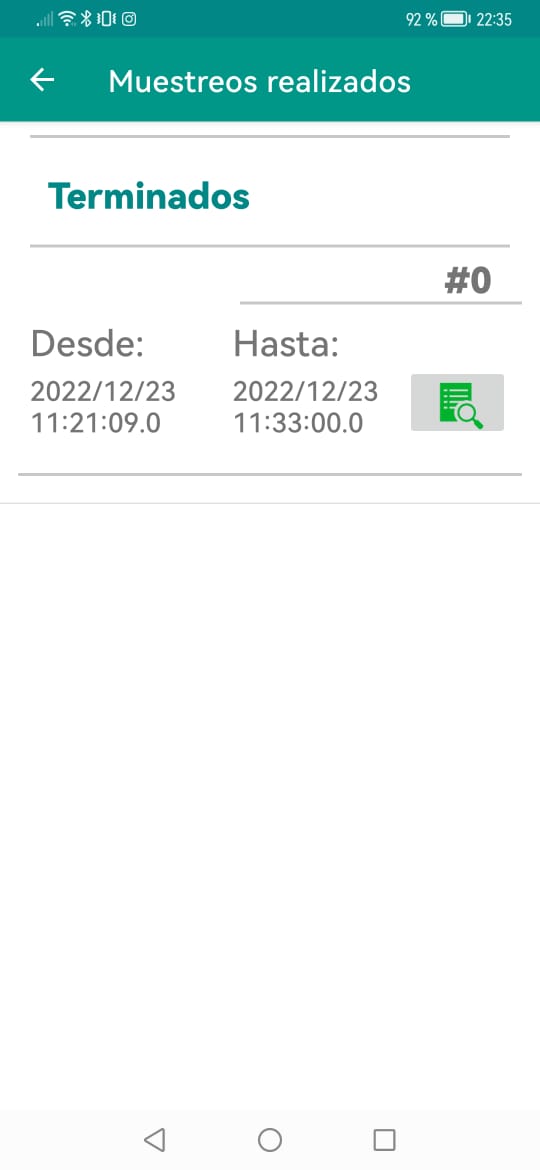
Adicionalmente, el usuario puede acceder a los registros de los distintos agendamientos al pulsar los tres puntos que son visibles desde cualquiera de los menús previamente mencionados, en el se presentan otras 2 opciones, la primera le permite acceder a los registros de cada dispositivo, la segunda opción permite visualizar los créditos de la aplicación.



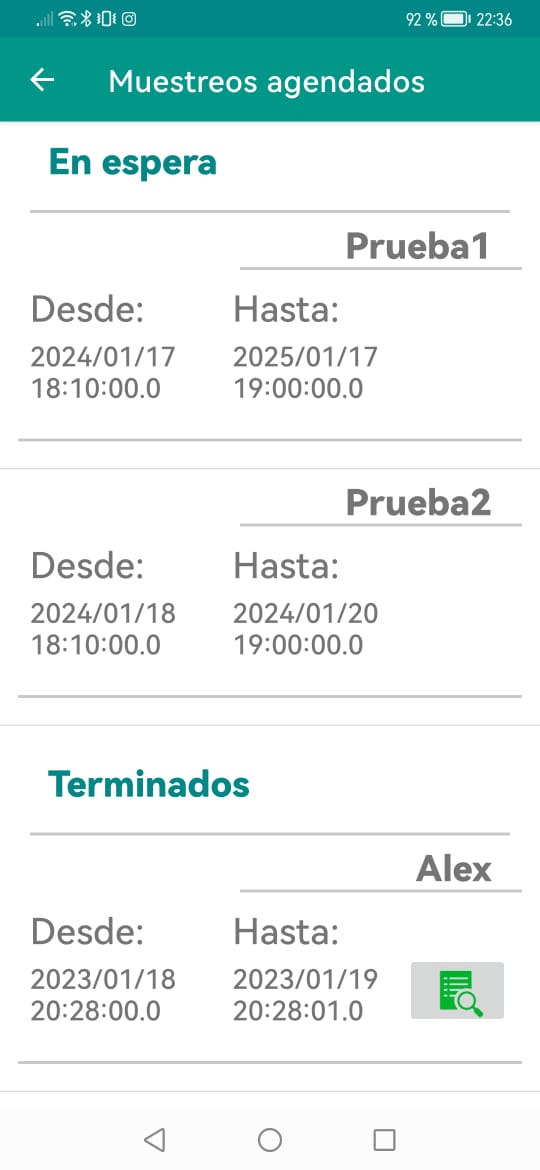
Al acceder a la primera opción el usuario es redirigido a una lista de los distintos dispositivos de donde se recopilo la información, y allí puede seleccionar el botón de inspeccionar (botón teal) para ver todos los muestreos realizados, o puede seleccionar el botón de agendamientos (botón morado) para acceder a todos los agendamientos del dispositivo,



En el menú de muestreos realizados (imagen de la izquierda) se puede ver la lista de muestreos ordenados por fechas, y se puede acceder a los datos de este al pulsar el botón de inspeccionar (botón verde). Este despliega la información del muestreo (imagen del medio), así como también ofrece la opción de exportar (imagen de la derecha).



Para la opción de agendamientos, esta despliega una lista con los distintos agendamientos del dispositivo, similar al menú de agendar, aquellos que están terminados pueden ser visualizados, mientras que los que están en espera pueden ser eliminados, sin embargo, para eliminarlos se requiere estar conectados al dispositivo, en las imágenes se puede ver que ocurre si no se está conectado.



# Código y repositorio

Como se mencionó en la introducción y con el objetivo de no extender la longitud del documento, el código completo, tanto como del ESP32, como de la aplicación se encuentra disponible en el siguiente repositorio de [GitHub](https://github.com/NikoGB/ESP32-OpticalEncoder-AndroidStudioProyect) para su disposicion, recomendamos comprobar la instalación de las distintas librerías y opciones presentadas en este documento en caso de presentar errores.