



ESPE
UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA



DEPARTAMENTO DE CIENCIAS DE LA ENERGÍA Y MECÁNICA
CARRERA DE INGENIERÍA MECATRÓNICA
TRABAJO DE TITULACIÓN, PREVIO A LA OBTENCIÓN DE:
INGENIERO MECATRÓNICO

“Diseño e implementación de un Sistema de Internet de las cosas (IoT) con realidad aumentada para el monitoreo de un banco de calibración de medidores de caudal de la empresa ESRW S.A. ”

AUTOR: QUINTERO TANGUILA, CARLOS ANDRÉS
DIRECTOR: ING. SINGAÑA AMAGUAÑA, MARCO ADOLFO

LATACUNGA, 2023



1. Introducción
2. Empresa
3. Planteamiento del problema
4. Objetivos
5. Hipótesis
6. Fundamentos Teóricos
7. Diseño e implementación
8. Pruebas y Resultados
9. Validación de hipótesis
10. Conclusiones y recomendaciones



En la actualidad el uso de las Tecnologías de la Información y la Comunicación (TIC), y especialmente el Internet de las Cosas (IoT), en la industria se han vuelto ineludibles, principalmente por ser vitales para incrementar la eficiencia organizacional y su nivel de competitividad. Esto ha impulsado la adopción de las TIC y la IoT en la mayoría de las actividades de la industria, lo que ha llevado al embrión de lo que es conocida como Industria 4.0 (también llamada Internet industrial de las cosas - IIoT).

La realidad aumentada se ha desarrollado como una de las tecnologías más importantes con referencia a obtener un apoyo didáctico, buscando de esta manera que un estudiante pueda desarrollar y utilizar un conocimiento sobre un fenómeno que pueda ser observado.



ESRAW S.A. es una empresa Servicios de Instrumentación que brinda servicios de Calibración, Inspección e Instrumentación para equipos de trabajo del sector industrial y petrolero, sus laboratorios se encuentran en la ciudad de El Coca.



PLANTEAMIENTO DEL PROBLEMA

La empresa ESRAW S.A., actualmente realiza procesos de calibración de instrumentos y equipos para el sector petrolero de la provincia de Orellana, los cuales son realizados por técnicos que realizan mediciones en puntos designados y posteriormente suben los datos obtenidos a una planilla. Debido a que realizan diferentes procesos de calibración, un técnico debe estar concentrado en qué magnitudes operan y cómo las evalúan, pero no están exentos a cometer errores, tanto al tomar mediciones como al ingresar los datos en la planilla, dado que si esto sucede puede ocasionar problemas tanto en la empresa cliente como en la empresa que calibra.



OBJETIVO GENERAL

Diseñar e implementar un IoT con realidad aumentada para la digitalización, monitoreo y generación de eventos de un banco de calibración de medidores de caudal de la empresa ESRAW S.A.



OBJETIVOS ESPECÍFICOS

- Recopilar información necesaria para la correcta maniobrabilidad del equipo por parte del estudiante.
- Investigar acerca de IoT, técnicas de simulación de objetos en realidad aumentada, su importancia y características.
- Seleccionar componentes para la implementación del sistema IoT con respecto a su mecánica y electrónica.
- Diseñar e implementar esquemas gráficos en 3D de los componentes, marcas de reconocimiento y aplicación de realidad aumentada.



OBJETIVOS ESPECÍFICOS

- Digitalizar y registrar variables del proceso: presión, temperatura, caudal y frecuencia del banco de calibración.
- Generar eventos a través de IoT: alarmas, arranque, paro del proceso.
- Implementar el software de monitoreo de datos del banco de calibración.
- Evaluar el funcionamiento de la aplicación en tiempo real del banco de calibración de medidores de caudal.
- Validar que la implementación del Sistema IoT con realidad aumentada permitirá la digitalización, monitoreo y generación de eventos de un banco de calibración de medidores de caudal de la empresa ESRAW S.A.



¿La implementación del Sistema Internet de las Cosas con realidad aumentada permitirá la digitalización, monitoreo y generación de eventos de un banco de calibración de medidores de caudal de la empresa ESRAW S.A.?



Internet de las cosas (IoT)

Es una red abierta e integral de objetos inteligentes que tienen la capacidad de autoorganizarse, compartir información, datos y recursos, reaccionando y actuando ante situaciones y cambios en el entorno.

Se considera que Internet de las cosas (IoT) es uno de los facilitadores de la próxima revolución industrial. Está impulsado por el avance de las tecnologías digitales, además de cambiar drásticamente la forma en que las empresas se involucran en las actividades comerciales y las personas interactúan con su entorno.



Realidad Aumentada (AR)

La realidad aumentada (AR) emerge como una aprendizaje inmersivo en la cual el entorno físico (RW) se enriquece mediante elementos generados por computadora que están asociados a ubicaciones o actividades específicas. En términos simples, AR posibilita que el contenido digital se integre de manera perfecta con nuestras percepciones del mundo real. Más allá de los objetos en 2D o 3D que se esperan comúnmente, los recursos digitales como archivos de audio y video, datos de texto e incluso información de carácter olfativo o táctil pueden fusionarse con la percepción del mundo real por parte de los usuarios.



Monitorización Industrial

Monitorización industrial también llamado seguimiento de procesos en entornos industriales, aprovecha las innovaciones tecnológicas actuales para supervisar el estado operativo de cualquier máquina o sistema. Mediante la incorporación de estos avances tecnológicos, es viable determinar si las máquinas están funcionando de manera adecuada y si se requiere llevar a cabo algún tipo de mantenimiento.



Google Firebase

Firebase es una empresa proveedora de BaaS (Backend como servicio) operada por Google.

Esencialmente, se trata de una base de datos en tiempo real, lo que significa que funciona como una base de datos NoSQL. Firebase almacena y organiza datos en una estructura jerárquica a través de un extenso árbol JSON.

Firebase engloba un abanico de prestaciones y utilidades valiosas para los programadores, que incluyen alternativas para la autenticación de usuarios, el almacenamiento en nube, la prestación de servicios web, la ejecución de funciones en la nube, la evaluación y supervisión de aplicaciones, así como la emisión de notificaciones push.



Google Apps Script

Google Apps Script (GAS) es una plataforma de desarrollo de secuencias de comandos creada por Google para la creación de aplicaciones ligeras en el entorno de Google. Mediante el uso de Google Apps Script, los desarrolladores tienen la capacidad de configurar conexiones externas para sistemas de administración de bases de datos relacionales o hojas de cálculo, permitiendo que estas últimas funcionen como una base de datos para sus aplicaciones.

Google Apps Script desempeña un papel fundamental al extender la funcionalidad de las hojas de cálculo de Google, permitiendo un mayor control y gestión de las mismas, así como su integración con otras aplicaciones y servicios.



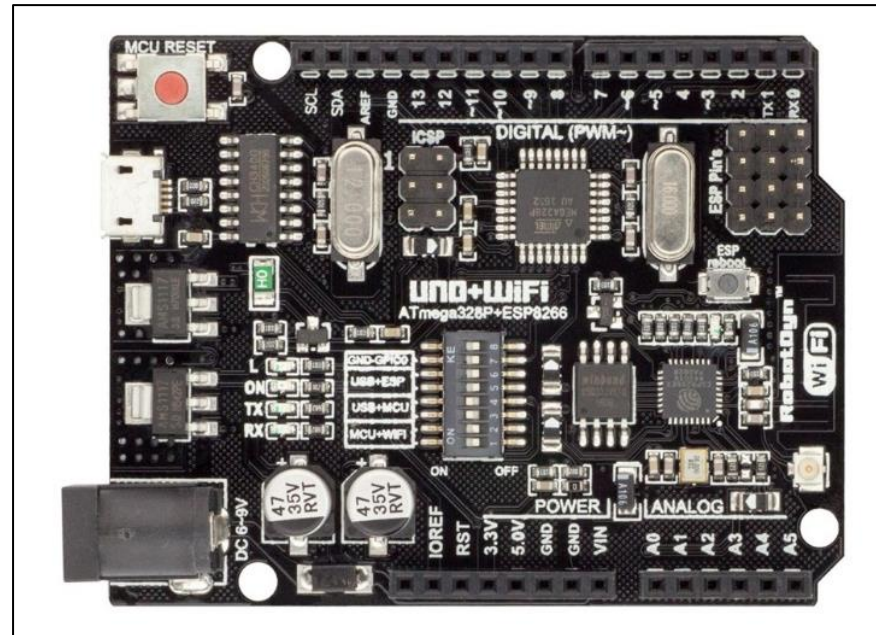
Google Looker Studio

Google Looker Studio, previamente conocida como Google Data Studio, se presenta como una aplicación de visualización de datos que proporciona una herramienta amigable para presentar conjuntos de datos complejos de manera atractiva y fácilmente comprensible. Este servicio se encuentra en la nube, permitiendo acceso desde cualquier ubicación, de manera gratuita, y posibilita la compartición de informes con seleccionados destinatarios.



Tarjeta UNO WiFi R3 ATmega328P ESP8266

Es una versión personalizada de la clásica placa ARDUINO UNO R3 que consiste en la integración completa del microcontrolador Atmel ATmega328 e IC WiFi ESP8266, con memoria flash de 32 MB y convertidor USB-TTL CH340G en una placa.



Metrología

La metrología abarca el dominio del saber referente a las mediciones, englobando todos los aspectos, tanto de índole teórica como práctica, que se vinculan con las mediciones, sin importar su nivel de precisión y en cualquier esfera de la ciencia y la tecnología. La meta de la metrología consiste en asegurar la uniformidad y confiabilidad de las mediciones, abarcando tanto las transacciones comerciales y de servicios, como los procesos industriales, así como los respectivos empeños de investigación científica y desarrollo tecnológico.



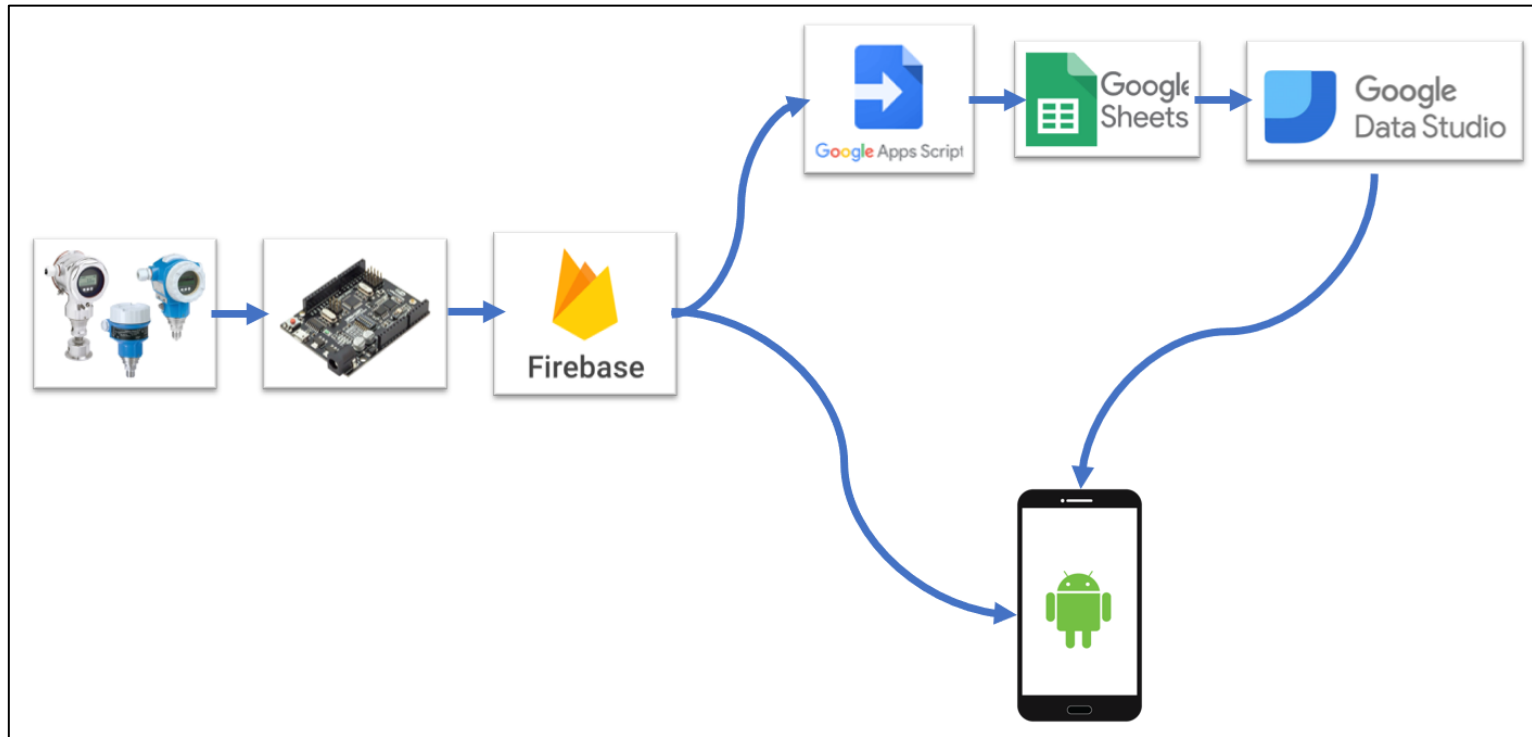
DISEÑO E IMPLEMENTACIÓN

El sistema de monitoreo en tiempo real se realizó mediante implementación de una tarjeta de adquisición de datos, una base de datos Firebase, una aplicación Android, Google Apps Script, Google Docs, Looker Studio y un complemento AR de Unity.

La aplicación Android se desarrolló para emitir notificaciones relevantes del banco de pruebas de medidores de caudal, mostrar los datos de los sensores en tiempo real, ver los registros de Looker Studio y observar los estados de los sensores mediante la integración de un complemento AR realizado en Unity.

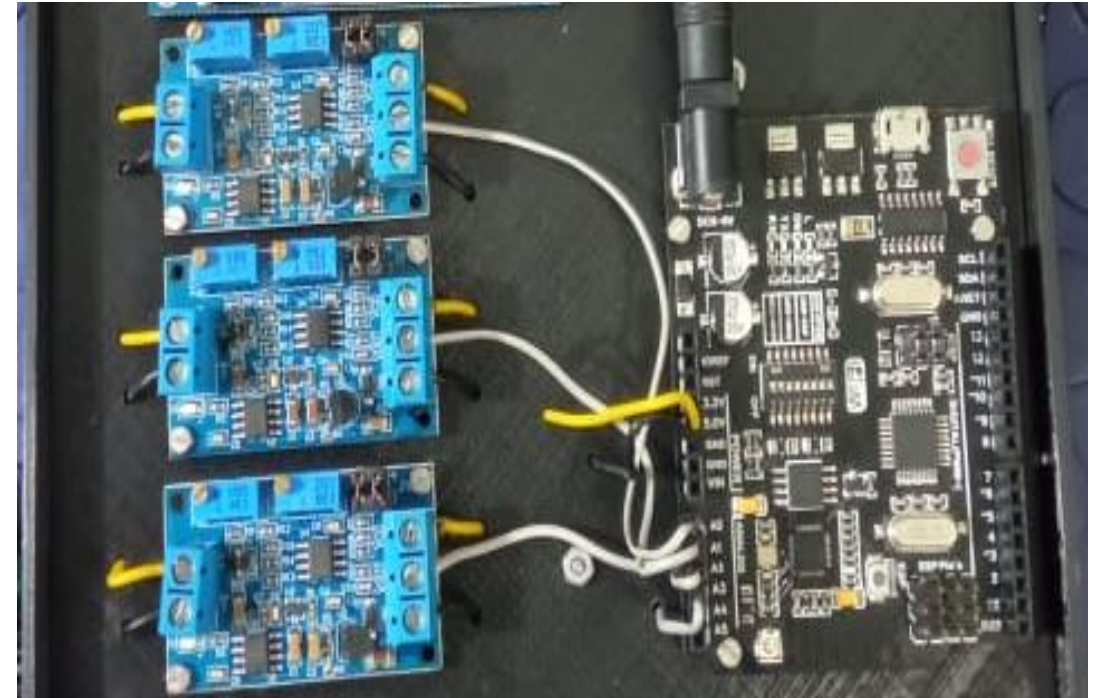
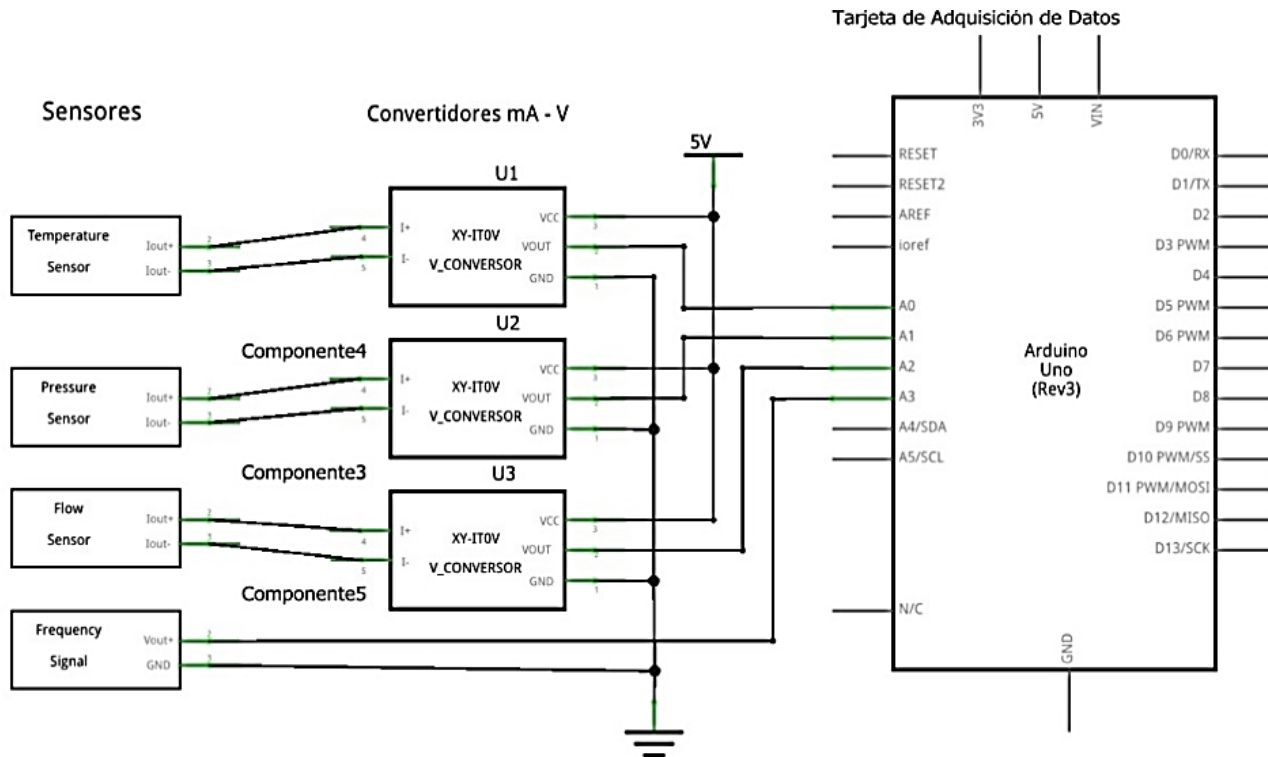


Diagrama de conexión



La tarjeta capta las medidas de los sensores del banco de pruebas y los envía a la base de datos, la aplicación android muestra la información de la base de datos y de los reportes de Google Data Studio.

Diagrama de conexión de las señales de sensores



DISEÑO E IMPLEMENTACIÓN

Características de los software y herramientas digitales utilizadas para el monitoreo

Software/Lenguaje	Características
Lenguaje de programación android	Java y Kotlin
Editor Android	Android Studio 4.1.1
Base de datos	Firebase
Google Scripts	JavaScript
Hoja de cálculo	Google Sheets
Registros y gráficos	DataStudio
Realidad Aumentada	Unity 2021.2.7f1
Programación Unity	C#
Arduino	Arduino 1.8+



Base de datos Firebase

Realtime Database

Datos Reglas Copias de seguridad Uso Extensiones NUEVA

<https://myloginapp-caqt.firebaseio.com>

<https://myloginapp-caqt.firebaseio.com/>

Estaciones

A: 1

B: 2

C: 3

D: 4

E: 5

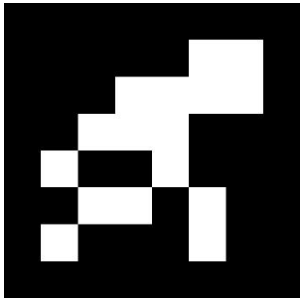
En el servicio de Realtime Database, en la parte de Datos, se agregan los campos que almacenarán los datos enviados por la tarjeta UNO WiFi R3

ATmega328P ESP8266, para este proyecto se denominó ESTACIONES, A-E.



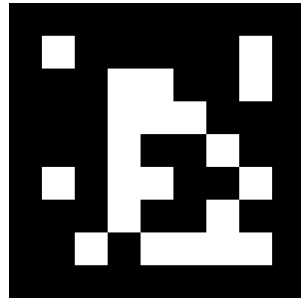
Marcadores Aruco para reconocimiento de sensores

ARUCO #33 de 6*6



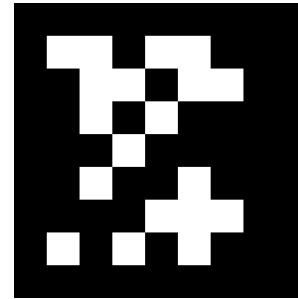
Marcador para
señal de
temperatura

ARUCO #91 de 7*7



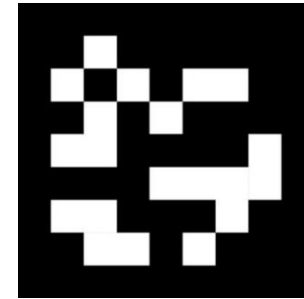
Marcador para
señal de presión

ARUCO #107 de 7*7



Marcador para
señal de caudal

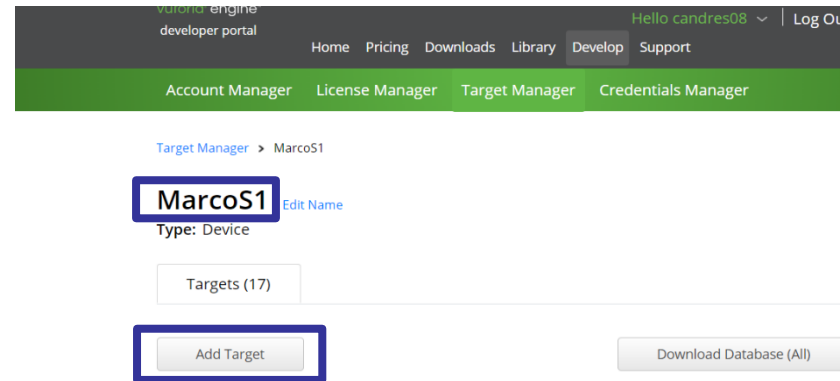
ARUCO #146 de 7*7



Marcador para
señal de
frecuencia

Base de datos de Vuforia

Los marcadores se deben añadir a una base de datos de Vuforia para poder implementarlo en Unity.



Add Target

Type:

Image Multi Cylinder Object

File:

7x7_1000-00.jpg Browse...

.jpg or .png (max file 2mb)

Width:

Enter the width of your target in scene units. The size of the target should be on the same scale as your augmented virtual content. Vuforia uses meters as the default unit scale. The target's height will be calculated when you upload your image.

Name:

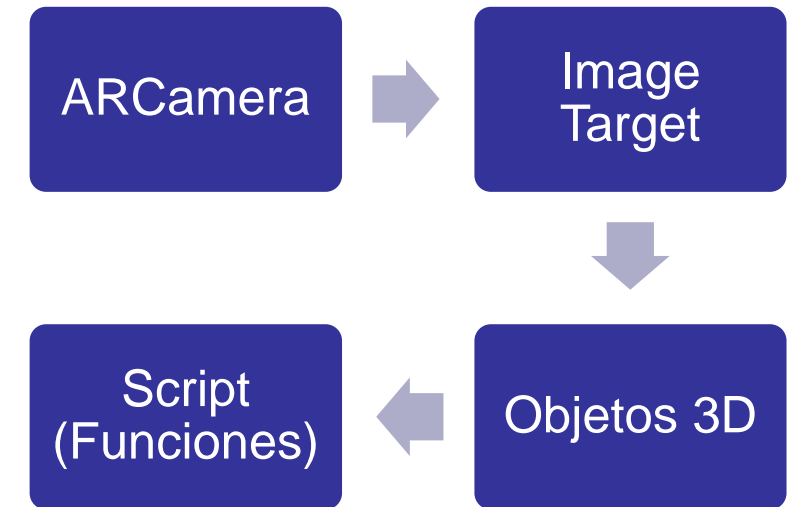
7x7_1000-00

Name must be unique to a database. When a target is detected in your application, this will be reported in the API.

Cancel Add



Creación de la herramienta AR



Funciones del script Database

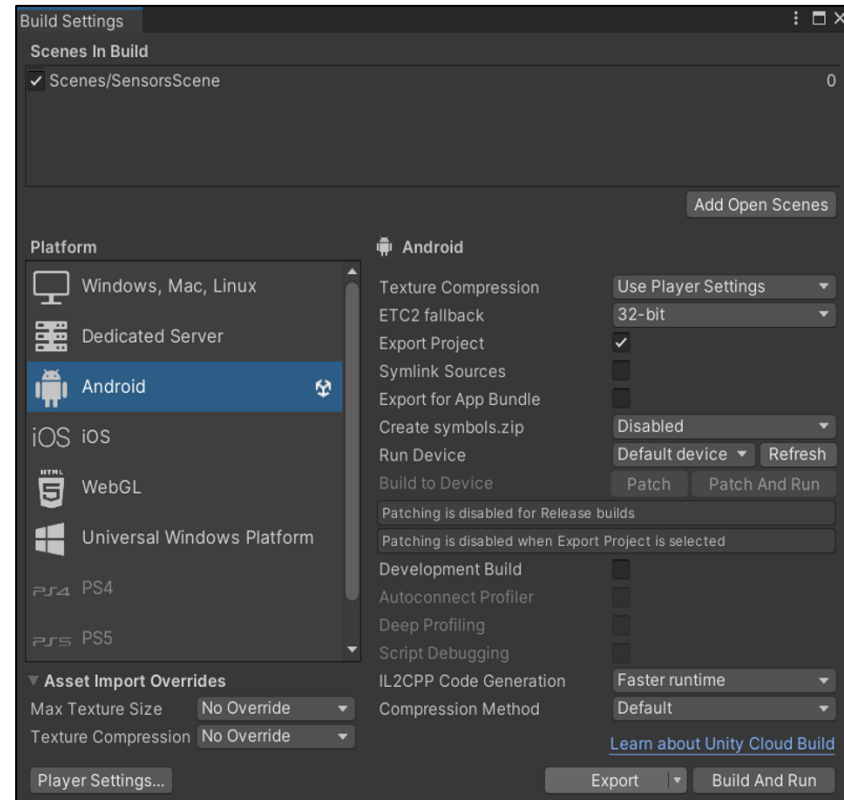
```
101 void NoLeerDatosPresion()  
102 {  
103     pres0.SetActive(false);  
104     pres1.SetActive(false);  
105     pres2.SetActive(false);  
106 }  
0 referencias  
107 void DatosPresion(string datoP)  
108 {  
109  
110     if (datoP == "OFF")  
111     {  
112         pres0.SetActive(true);  
113         pres1.SetActive(false);  
114         pres2.SetActive(false);  
115     }  
116     else if (datoP == "ON")  
117     {  
118         pres0.SetActive(false);  
119         pres1.SetActive(true);  
120         pres2.SetActive(false);  
121     }  
122     else if (datoP == "FAIL")  
123     {  
124         pres0.SetActive(false);  
125         pres1.SetActive(false);  
126         pres2.SetActive(true);  
127     }  
128     text.text = "Presion";  
129 }
```

Son funciones paramétricas, en las que ingresan un dato desde la aplicación android.

Asignan que objeto 3D debe activarse cuando reconozca el marcador el *Image Target*.



Exportación de la herramienta AR



Programación de la Tarjeta UNO WiFi R3 ATmega328P ESP8266

Se crea dos programas en Arduino para recibir del banco de pruebas y enviarlos a la base de datos de Firebase.

```
ReceptordeDatos Arduino 1.8.13
Archivo Editar Programa Herramientas Ayuda

ReceptordeDatos $
27 //Se realiza la lectura del sensor 1
28 valorSensor1 = analogRead(sensor1);
29 //cambia la escala a 0.0 - 100.0
30 vSensor1 = map(valorSensor1, 0, 204.6, 0.0, 250);
31 vSensor1=vSensor1/100;
32 //Se realiza la lectura del sensor 2
33 valorSensor2 = analogRead(sensor2);
34 //cambia la escala a 0.0 - 100.0
35 vSensor2 = map(valorSensor2, 0, 368, 0, 2350);
36 vSensor2=vSensor2/100;
37 //Se realiza la lectura del sensor 3
38 valorSensor3 = analogRead(sensor3);
39 //cambia la escala a 0.0 - 100.0
40 vSensor3 = map(valorSensor3, 0, 409, 0.0, 10000);
41 vSensor3=vSensor3/100;
42 //Se realiza la lectura del sensor 4
43 valorSensor4 = analogRead(sensor4);
44 //cambia la escala a 0.0 - 100.0
45 vSensor4 = map(valorSensor4, 0, 1084, 0.0, 6000);
46 vSensor4=vSensor4/100;
47 // Serial.println(vSensor4);
48
49 //Enviamos los valores de los sensores por comunicación serial al ESP8266 para su posterior env
50 //Preparación para enviar valores de los sensores por serial al esp8266
51 DynamicJsonBuffer jbuffer;
52 JsonObject& root = jbuffer.createObject();
53 root["Sensor1"] = String(vSensor1); // configurando la variable Sensor1 para contener el valor
54 root["Sensor2"] = String(vSensor2); // configurando la variable Sensor2 para contener el valor
55 root["Sensor3"] = String(vSensor3); // configurando la variable Sensor3 para contener el valor
56 root["Sensor4"] = String(vSensor4); // configurando la variable Sensor4 para contener el valor
57 root.printTo (Serial); // enviando los valores de los sensores por serial al esp8266
58 root.remove ("Sensor1"); // elimina el campo temporal del objeto raíz JSON para poder reutiliza
59 root.remove ("Sensor2"); // elimina el campo temporal del objeto raíz JSON para poder reutiliza
60 root.remove ("Sensor3"); // elimina el campo temporal del objeto raíz JSON para poder reutiliza
61 root.remove ("Sensor4"); // elimina el campo temporal del objeto raíz JSON para poder reutiliza
62 delay(2000);
63 }
```

```
Firestore_ESP8266 Arduino 1.8.13
Archivo Editar Programa Herramientas Ayuda

Firestore_ESP8266
40 // En el ciclo loop se desarrolla el código que se va a estar repitiendo constantemente
41 void loop() {
42 // Definición de variables para leer los valores del Arduino
43
44 while(1){
45 // Aceptar datos del arduino
46 while (Serial.available()) { // Leer datos en serie del Arduino
47 json = Serial.readString();
48 StringReady = true;
49 }
50 if (StringReady) {
51 StaticJsonBuffer <200> jsonBuffer; // memoria preasignada para almacenar el JsonObject max 200 bytes
52 JsonObject & root = jsonBuffer.parseObject (json); // convertir la cadena en datos JSON
53
54 if (!root.success()) {
55 Serial.println ("parseObject () failed");
56 return;
57 }
58 //Declaración de variables para almacenar los valores leído del arduino
59 String rawSensor1 = root["Sensor1"]; //
60 String rawSensor2 = root["Sensor2"]; //
61 String rawSensor3 = root["Sensor3"]; //
62 String rawSensor4 = root["Sensor4"]; //
63 // Conversión en número flotantes a los valores obtenidos
64 vSensor1=rawSensor1.toFloat();
65 vSensor2=rawSensor2.toFloat();
66 vSensor3=rawSensor3.toFloat();
67 vSensor4=rawSensor4.toFloat();
68 root.remove ("Sensor1"); // elimina el campo temporal del objeto raíz JSON para poder reutilizar el ob
69 root.remove ("Sensor2"); // elimina el campo temporal del objeto raíz JSON para poder reutilizar el ob
70 root.remove ("Sensor3"); // elimina el campo temporal del objeto raíz JSON para poder reutilizar el ob
71 root.remove ("Sensor4"); // elimina el campo temporal del objeto raíz JSON para poder reutilizar el ob
72 }
73
74 /// Programación para enviar a la base de datos
75
76 //Si el valor del sensor ha cambiado respecto a la lectura previa actualiza el valor en la base de dat
77 }
```



Implementación de Firebase en aplicación Android

Es necesario crear un proyecto de Firebase y posteriormente implementarlo en la aplicación Android para que se habiliten los servicios de autenticación y de base de datos.

The image displays two screenshots from the Firebase console and Android Studio, illustrating the process of adding Firebase to an Android application.

Left Screenshot (Firebase Console): Shows the "Agrega Firebase a tu app para Android" (Add Firebase to your Android app) page. Step 1, "Registrar app" (Register app), is active. The "Nombre del paquete de Android" (Android package name) is set to "com.unity.mynativeapp" and the "Sobrenombre de la app (opcional)" (Optional app nickname) is "MyLoginApp". The "Certificado de firma SHA-1 de depuración (opcional)" (Optional debug signing certificate SHA-1) is "58:EF:62:74:F5:30:A9:06:84:2D:CF:98:C6:0A:12:6B:96". A blue box highlights the "Registrar app" button.

Right Screenshot (Android Studio): Shows the "Agrega Firebase a tu app para Android" page. Step 2, "Descargar y, luego, agregar el archivo de configuración" (Download and then add the configuration file), is active. A blue box highlights the "Descargar google-services.json" (Download google-services.json) button. Below, a blue box highlights the "Siguiete" (Next) button. A blue arrow points from the "Descargar google-services.json" button to the "Project" view in Android Studio, which shows the "google-services.json" file being added to the root of the module.

Splash Screen

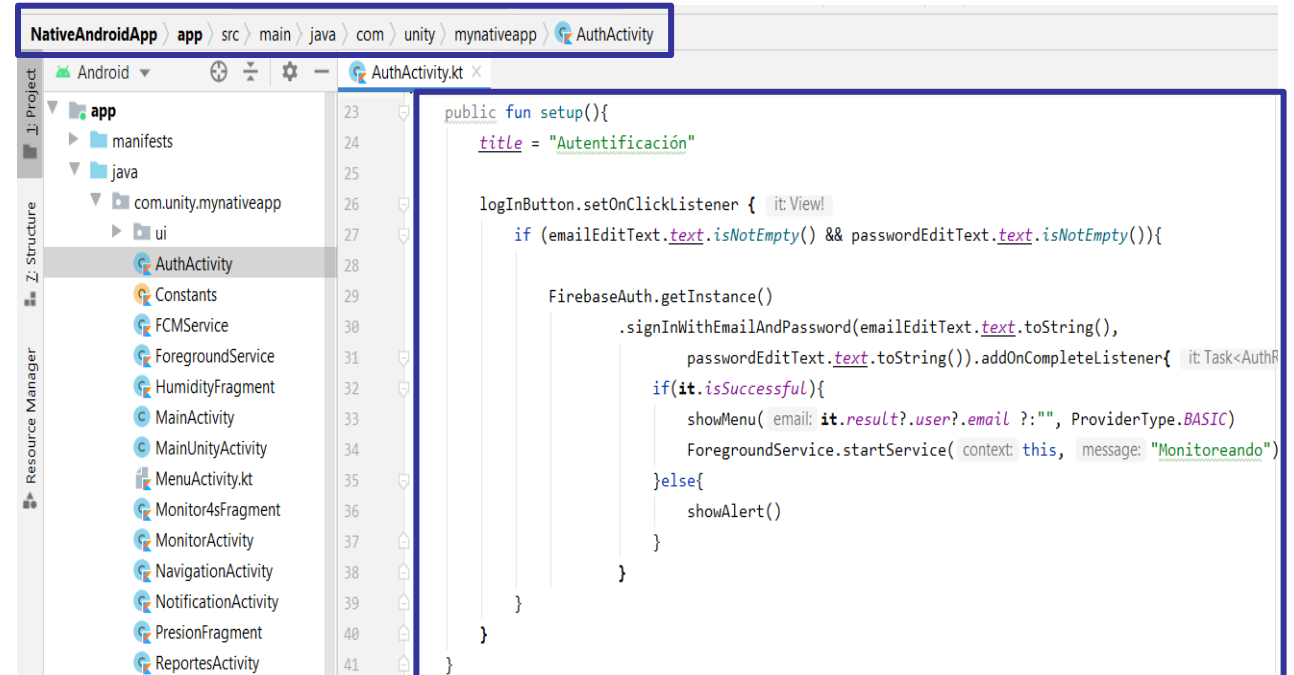
Se implementa un splash screen como pantalla de carga para que al iniciar la aplicación se vea el nombre y logo de la empresa.



```
NativeAndroidApp > app > src > main > java > com > unity > mynativeapp > AuthActivity
Android
  app
    manifests
    AndroidManifest.xml
    java
      com.unity.mynativeapp
        ui
          AuthActivity
          Constants
          FCMSERVICE
          ForegroundService
          HumidityFragment
          MainActivity
          MainUnityActivity
          MenuActivity.kt
          Monitor4sFragment
          MonitorActivity
          NavigationActivity
AuthActivity.kt
AndroidManifest.xml
5  import androidx.appcompat.app.AlertDialog
6  import androidx.appcompat.app.AppCompatActivity
7  import com.google.firebase.auth.FirebaseAuth
8  import kotlinx.android.synthetic.main.activity_auth.*
9
10
11 class AuthActivity : AppCompatActivity() {
12     override fun onCreate(savedInstanceState: Bundle?) {
13         //Splash Screen
14         Thread.sleep( millis: 1000)
15         setTheme(R.style.AppTheme)
16
17         super.onCreate(savedInstanceState)
18         setContentView(R.layout.activity_auth)
19
20         setup()
21     }
```

Programación para Autenticar usuarios

El código autentica al usuario mediante Firebase Authentication utilizando un correo electrónico y contraseña proporcionados. Si la autenticación tiene éxito, se muestra la pantalla de Menú y se inicia un servicio en primer plano para emitir notificaciones, si la autenticación falla, se muestra una alerta.

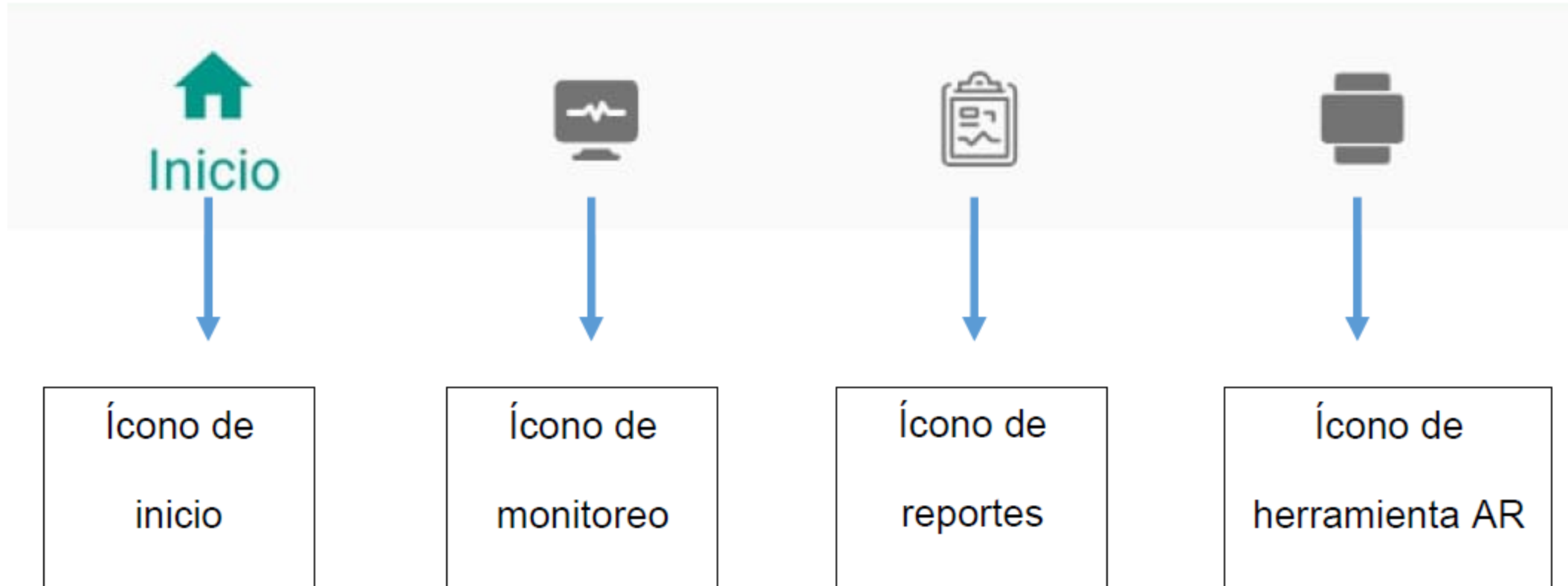


```
public fun setup(){
    title = "Autenticación"

    loginButton.setOnClickListener { it: View!
        if (emailEditText.text.isNotEmpty() && passwordEditText.text.isNotEmpty()){

            FirebaseAuth.getInstance()
                .signInWithEmailAndPassword(emailEditText.text.toString(),
                    passwordEditText.text.toString()).addOnCompleteListener{ it: Task<AuthR
                if(it.isSuccessful){
                    showMenu( email: it.result?.user?.email ?: "", ProviderType.BASIC)
                    ForegroundService.startService( context: this, message: "Monitoreando")
                }else{
                    showAlert()
                }
            }
        }
    }
}
```

Barra de navegación



Codificación de la ventana Monitoreo

Este código establece una función que lee la base de datos de Firebase. Cada vez que los datos en la base de datos cambian, la función “onDataChange” se ejecuta, obteniendo los valores de diferentes nodos y actualizando las vistas en la interfaz de usuario con esos valores.

The image shows a side-by-side view of a code editor and a mobile application interface. The code editor on the left displays the following Kotlin code for `Monitor4sFragment`:

```
28 }
29
30 override fun onCreate(savedInstanceState: Bundle?) {
31     super.onCreate(savedInstanceState)
32
33     val database = Firebase.database
34     val myRef = database.reference
35
36     myRef.addValueEventListener(object : ValueEventListener {
37         @SuppressLint("SetTextI18n")
38         override fun onDataChange(dataSnapshot: DataSnapshot) {
39             val temperaturav: String = dataSnapshot.child( path: "Estaciones/B").value.toString()
40             val presionv: String = dataSnapshot.child( path: "Estaciones/C").value.toString()
41             val flujov: String = dataSnapshot.child( path: "Estaciones/D").value.toString()
42             val frecuenciav: String = dataSnapshot.child( path: "Estaciones/E").value.toString()
43
44             temperatureTextView.text = temperaturav + " C°"
45             presionTextView.text = presionv + " psi"
46             flujoTextView.text = flujov + " gal/min"
47             turbinaTextView.text = frecuenciav + " Hz"
48         }
49     })
50
51     override fun onCancelled(error: DatabaseError) {
52         Log.w( tag: "TAG", msg: "Failed to read value.", error.toException())
53     }
54 }
55 }
```

The mobile app interface on the right shows a real-time monitoring screen titled "Monitoreo en tiempo real". It displays data for "Monitoreo del Banco de Calibración" with the following values:

Temperatura	Presión
26.1 C°	2.5 psi
Frecuencia	Caudal
35.1 Hz	136.1 gal/min

DISEÑO E IMPLEMENTACIÓN

Codificación de la ventana Registros

Se programa una ventana que sirva con WebView, para poder visualizar un dirección web, que será la de Google Looker Studio en la cual se creo el registro.

The image displays the development environment for an Android application. On the left, the Project and Resource Manager panes show the file structure, with 'RegistroActivity' selected. The central pane shows the Kotlin code for 'RegistroActivity', which implements a swipe-to-refresh feature and a WebView to display data from Google Looker Studio. The code includes logic for refreshing the page and handling URL loading.

```
35 //Refresh
36 swipeRefresh.setOnRefreshListener {
37     webView.reload()
38 }
39
40 swipeRefresh.isEnabled = false
41 webView.settings.javaScriptEnabled = true
42 webView.settings.domStorageEnabled = true
43
44 //WebView
45 webView.webChromeClient = object : WebChromeClient() {
46 }
47 webView.webViewClient = object : WebViewClient() {
48     override fun shouldOverrideUrlLoading(view: WebView?, url: String?): Boolean {
49         return false
50     }
51     override fun onPageStarted(view: WebView?, url: String?, favicon: Bitmap?) {
52         super.onPageStarted(view, url, favicon)
53         swipeRefresh.isRefreshing = true
54     }
55     override fun onPageFinished(view: WebView?, url: String?) {
56         super.onPageFinished(view, url)
57         swipeRefresh.isRefreshing = false
58     }
59 }
60
61 webView.refreshStateListener = { it: Boolean
62     swipeRefresh.isEnabled = it
63 }
64
65 webView.loadUrl(BASE_URL)
```

On the right, a mobile app interface titled 'Reportes' is shown. It features a date range selector for '1 ene 2022 - 1 ene 2022' and a table of data. Below the table is a line chart with four series: Presión (blue), Temperatura (green), Flujo (purple), and Frecuencia (orange). The chart shows data points over time, with the x-axis labeled with dates and times.

Hora	Presión	Temperatura	Flujo	Frecuencia	Fecha
1. 2:27:46 a.m.	7.11	23	9	4.5	20220101...
2. 4:27:46 a.m.	7.11	23	9	4.5	20220101...
3. 5:27:46 a.m.	7.11	23	9	4.5	20220101...
4. 6:27:46 a.m.	7.11	23	9	4.5	20220101...
5. 8:27:46 a.m.	7.11	23	9	4.5	20220101...
6. 9:27:46 a.m.	7.11	23	9	4.5	20220101...
7. 10:27:46 a.m.	7.11	23	9	4.5	20220101...
8. 11:27:46 a.m.	7.11	23	9	4.5	20220101...
9. 12:28:02 p.m.	7.11	23	9	4.5	20220101...
10. 1:27:46 p.m.	7.11	23	9	4.5	20220101...
11. 2:27:46 p.m.	7.11	23	9	4.5	20220101...
12. 4:27:47 p.m.	7.11	23	9	4.5	20220101...
13. 5:27:46 p.m.	7.11	23	9	4.5	20220101...



Registro de datos en Looker Studio

The screenshot shows a Looker Studio report interface. The main content area displays a table with the following data:

	Hora	Presión	Temperatura	Flujo	Frecuencia	Fecha
1.	6:42:48 p. m.	44,45	0,95	85,79	22,48	20220922...
2.	12:25:02 p. m.	25,3	5,5	171,58	41,9	20220805...
3.	8:08:11 a. m.	25,3	5,5	171,58	41,9	20220806...
4.	9:14:48 p. m.	25,3	5,5	171,58	41,9	20220731...
5.	12:42:25 a. m.	25,3	5,5	171,58	41,9	20220801...
6.	9:46:49 a. m.	25,3	5,5	171,58	41,9	20220807...
7.	9:33:50 a. m.	25,3	5,5	171,58	41,9	20220729...
8.	8:24:50 p. m.	25,3	5,5	171,58	41,9	20220806...
9.	3:56:49 p. m.	25,3	5,5	171,58	41,9	20220727...
10.	4:42:48 a. m.	25,3	5,5	171,58	41,9	20220803...
11.	1:51:51 a. m.	25,3	5,5	171,58	41,9	20220727...
12.	7:31:48 p. m.	25,3	5,5	171,58	41,9	20220829...
13.	11:38:12 p. m.	25,3	5,5	171,58	41,9	20220807...

The interface also includes a 'Datos' panel on the right with a search bar and a list of data fields: Contador, Estado, Fecha, Flujo, Frecuencia, Hora, Presión, Temperatura, U Fecha, U Hora, U SignalA, U SignalB, U SignalC, U SignalD, U SignalE, and Record Count. Below the list are buttons for 'Añadir un campo', 'Añadir un parámetro', and 'Añadir datos'. The main report area has a blue header 'BANCO MEDIDOR DE CAUDAL' and a dropdown menu 'Selecciona un periodo'.



Programación en Google Scripts

Obtiene los datos de la base de datos Firebase.

```
function obtenerDatos1() {
  // Dirección de la base de datos RealTime de Firebase
  var bd = "myloginapp-caqt.firebaseio.com" ;
  // Secreto de la base de datos Firebase
  var secreto = 'WBWK69LIDV1BwEQxk6bBxPtPOFswuehhIbj18poy';
  // Ruta de la base de datos RealTime
  var ruta = "Estaciones";

  var options = { method:'get', contentType: 'application/json' }; //opciones de la llamada a fetch
  var firebaseURL = 'https://'+bd+'/'+ruta+'.json?auth='+secreto; // construyo la ruta

  var resultado = UrlFetchApp.fetch(firebaseURL, options);

  // Hoja de cálculo activa
  var ss = SpreadsheetApp.getActive();
  var sheet = ss.getSheetByName("portafolio")

  var ultimalineas = sheet.getLastRow(); // obtiene la última línea

  // Fechas y horas actuales.
  var fecha=Utilities.formatDate(new Date, "GMT-5", "yyyy/MM/dd");
  var d = new Date();
  var hora = d.toLocaleTimeString();
}
```

Envío de datos a la hoja de cálculo de Google

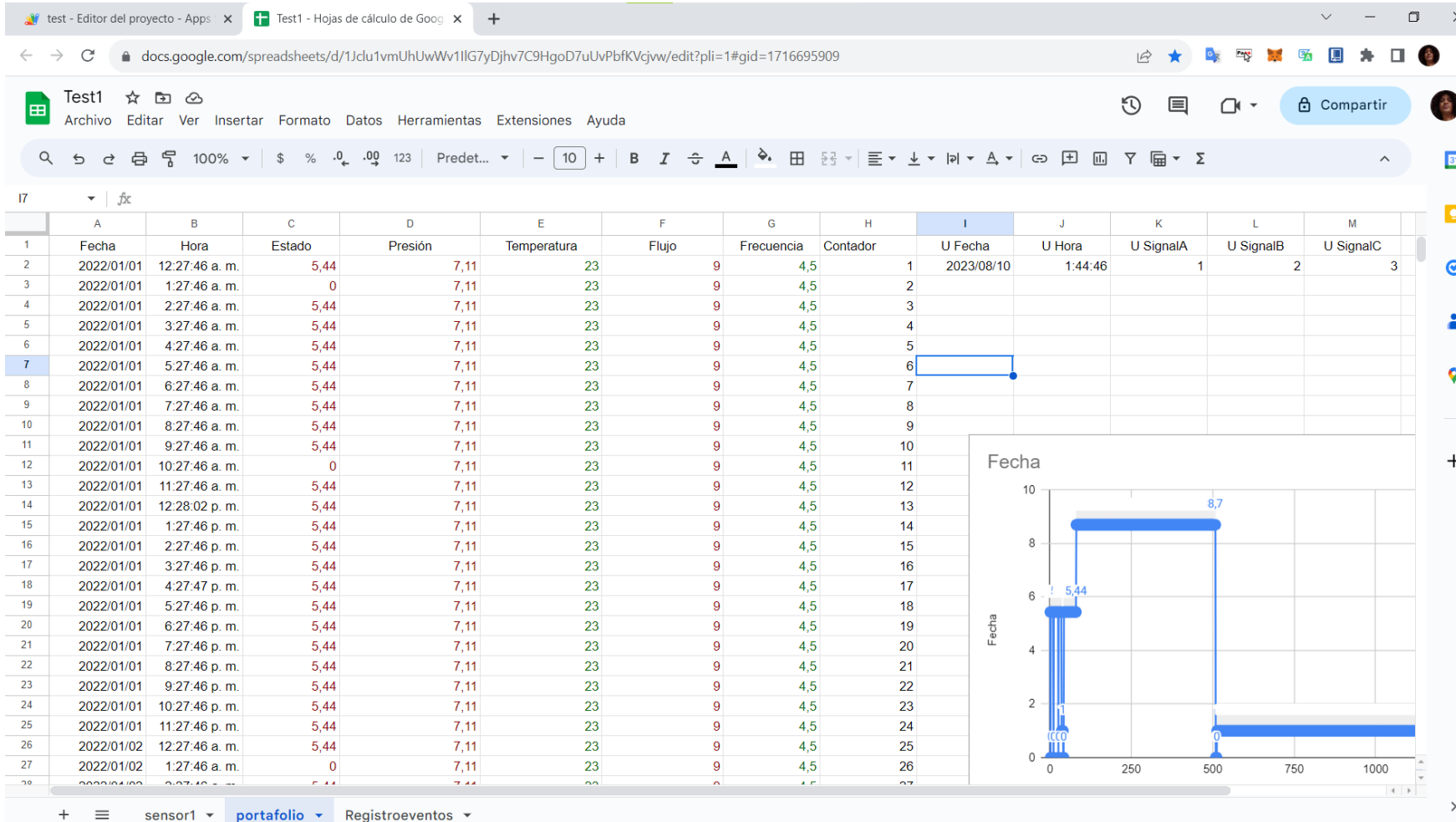
```
// Objeto JSON según el resultado
var e = JSON.parse(resultado);
//var e = importDate();
//Logger.log(e["A"]);
//if(e["A"]>0){
if(true){

  for (clave in e) {
    var columna = clave.charCodeAt(0)-62; // columna de cada valor
    //Logger.log(columna);
    var valorActual = sheet.getRange(2,columna).getValue();
    if (true) { // Si los valores son distintos
      // 1) Cambio fecha y hora en la fila dos
      sheet.getRange(2,9).setValue(fecha);
      sheet.getRange(2,10).setValue(hora);
      // 2) Cambio valor actual en la fila dos
      sheet.getRange(2,columna+8).setValue((e[clave]));
      // 3) Pongo fecha y hora en la última línea
      sheet.getRange(ultimalineas+1,1).setValue(fecha);
      sheet.getRange(ultimalineas+1,2).setValue(hora);
      // 4) Escribo en la última línea un registro de lo que acontece
      segunActivo1(clave,e[clave],ultimalineas+1,columna,sheet);
    }else{
      blanco(ultimalineas+1,columna,sheet);
    }

    // Logging de los datos...
    Logger.log(e);
    Logger.log(clave);
    //Logger.log(e[clave]);
  }
}
```



Hoja de Cálculo de Google



Codificación de la ventana Monitoreo

Este código establece una función que lee la base de datos de Firebase. Cada vez que los datos en la base de datos cambian, la función “onDataChange” se ejecuta, obteniendo los valores de diferentes nodos y actualizando las vistas en la interfaz de usuario con esos valores.

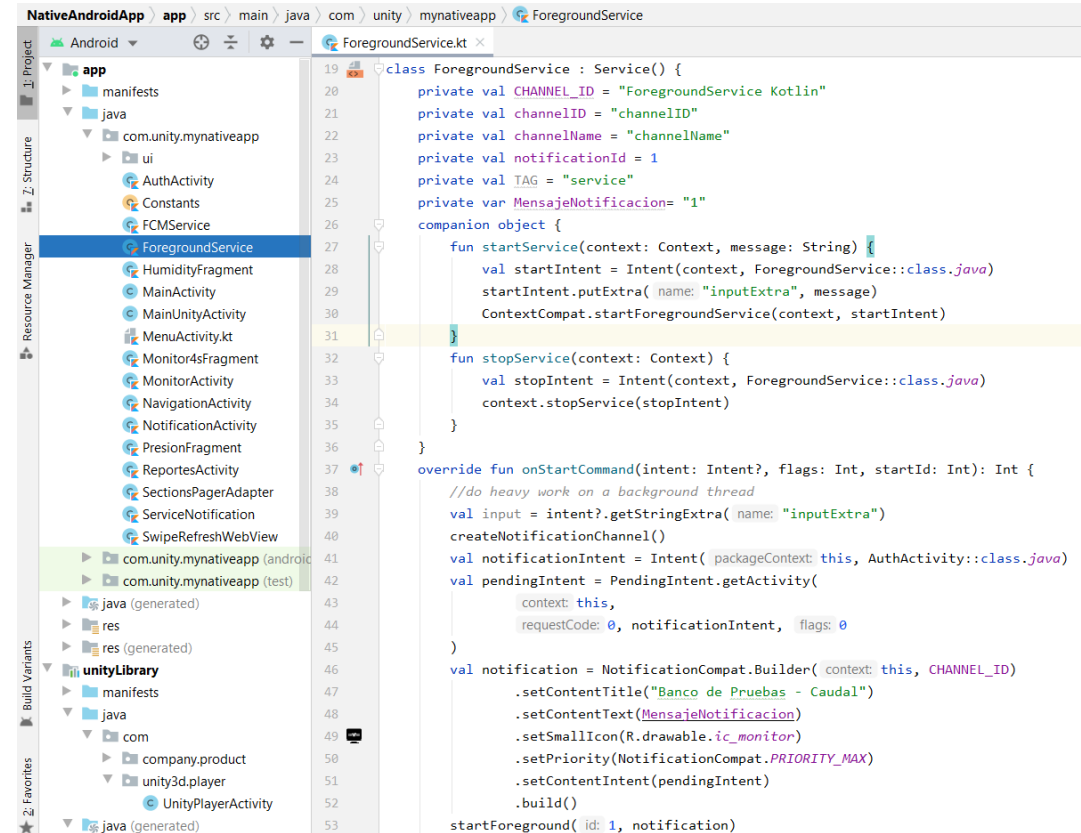
The image displays two side-by-side views. On the left is the Android Studio IDE showing the Kotlin code for the `Monitor4sFragment` class. The code implements the `onActivityCreated` method to initialize a Firebase database reference and the `onDataChange` method to listen for data updates and update UI text views with values for temperature, pressure, flow, and frequency. On the right is a screenshot of the mobile application interface. The app has a green header with the text "Monitoreo en tiempo real". Below the header, it displays "Monitoreo del Banco de Calibración" and "En tiempo real". The main content area shows two rows of data: "Temperatura" (26.1 C°) and "Presión" (2.5 psi) in the first row, and "Frecuencia" (35.1 Hz) and "Caudal" (136.1 gal/min) in the second row. The bottom navigation bar includes a home icon, a "Monitoreo" tab, and other icons.

```
com > unity > mynativeapp > Monitor4sFragment
Monitor4sFragment.kt
28 }
29
30 override fun onActivityCreated(savedInstanceState: Bundle?) {
31     super.onActivityCreated(savedInstanceState)
32
33     val database = Firebase.database
34     val myRef = database.reference
35
36     myRef.addValueEventListener(object : ValueEventListener {
37         @SuppressWarnings("SetTextI18n")
38         override fun onDataChange(dataSnapshot: DataSnapshot) {
39             val temperaturav: String = dataSnapshot.child( path: "Estaciones/B").value.toString()
40             val presionv: String = dataSnapshot.child( path: "Estaciones/C").value.toString()
41             val flujov: String = dataSnapshot.child( path: "Estaciones/D").value.toString()
42             val frecuenciav: String = dataSnapshot.child( path: "Estaciones/E").value.toString()
43
44             temperatureTextView.text = temperaturav + " C°"
45             presionTextView.text = presionv + " psi"
46             flujoTextView.text = flujov + " gal/min"
47             turbinaTextView.text = frecuenciav + " Hz"
48         }
49
50         override fun onCancelled(error: DatabaseError) {
51             Log.w( tag: "TAG", msg: "Failed to read value.", error.toException())
52         }
53     })
54 }
55 }
```



Servicio Foreground

Se crea un servicio en primer plano para que pueda ejecutar la notificaciones aun cuando la aplicación se encuentra en segundo plano.



```
NativeAndroidApp > app > src > main > java > com > unity > mynativeapp > ForegroundService
class ForegroundService : Service() {
    private val CHANNEL_ID = "ForegroundService Kotlin"
    private val channelId = "channelID"
    private val channelName = "channelName"
    private val notificationId = 1
    private val TAG = "service"
    private var MensajeNotificacion= "1"
    companion object {
        fun startService(context: Context, message: String) {
            val startIntent = Intent(context, ForegroundService::class.java)
            startIntent.putExtra(name: "inputExtra", message)
            ContextCompat.startForegroundService(context, startIntent)
        }
        fun stopService(context: Context) {
            val stopIntent = Intent(context, ForegroundService::class.java)
            context.stopService(stopIntent)
        }
    }
    override fun onStartCommand(intent: Intent?, flags: Int, startId: Int): Int {
        //do heavy work on a background thread
        val input = intent?.getStringExtra(name: "inputExtra")
        createNotificationChannel()
        val notificationIntent = Intent(packageContext: this, AuthActivity::class.java)
        val pendingIntent = PendingIntent.getActivity(
            context: this,
            requestCode: 0, notificationIntent, flags: 0
        )
        val notification = NotificationCompat.Builder(context: this, CHANNEL_ID)
            .setContentTitle("Banco de Pruebas - Caudal")
            .setContentText(MensajeNotificacion)
            .setSmallIcon(R.drawable.ic_monitor)
            .setPriority(NotificationCompat.PRIORITY_MAX)
            .setContentIntent(pendingIntent)
            .build()
        startForeground(id: 1, notification)
    }
}
```



Programación para enviar datos de la herramienta AR (librería unity)

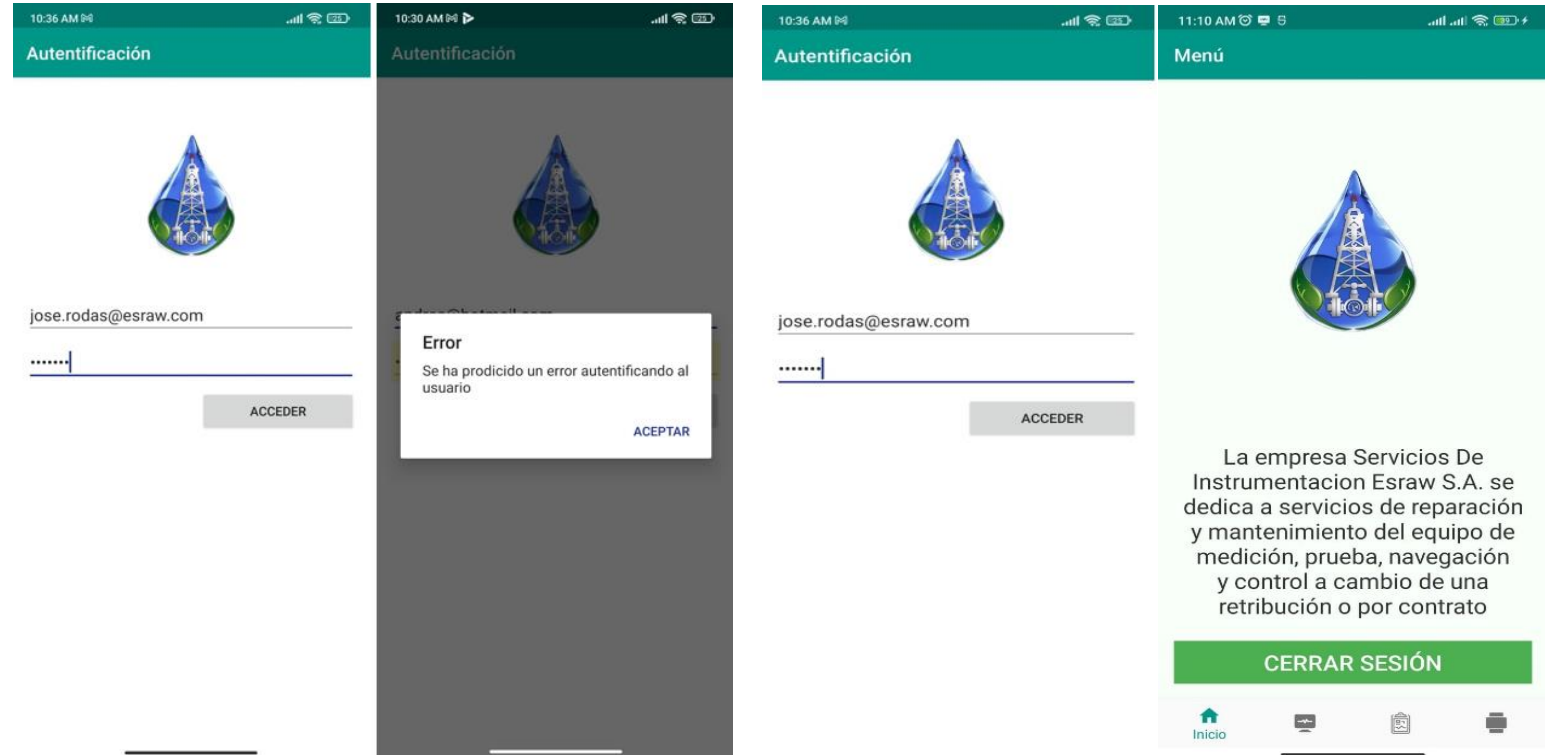
Se accede al script Database de la librería de unity mediante las líneas de código de la imagen en la cual se ingresan como parámetros el nombre del script, el nombre de la variable del script y el valor del estado del de cada sensor.

```
mUnityPlayer.UnitySendMessage( s: "Database", s1: "DatosTemp", DatoT);  
mUnityPlayer.UnitySendMessage( s: "Database", s1: "DatosPresion", DatoP);  
mUnityPlayer.UnitySendMessage( s: "Database", s1: "DatosCaudal", DatoC);  
mUnityPlayer.UnitySendMessage( s: "Database", s1: "DatosFrec", DatoF);
```



Autenticación de usuario

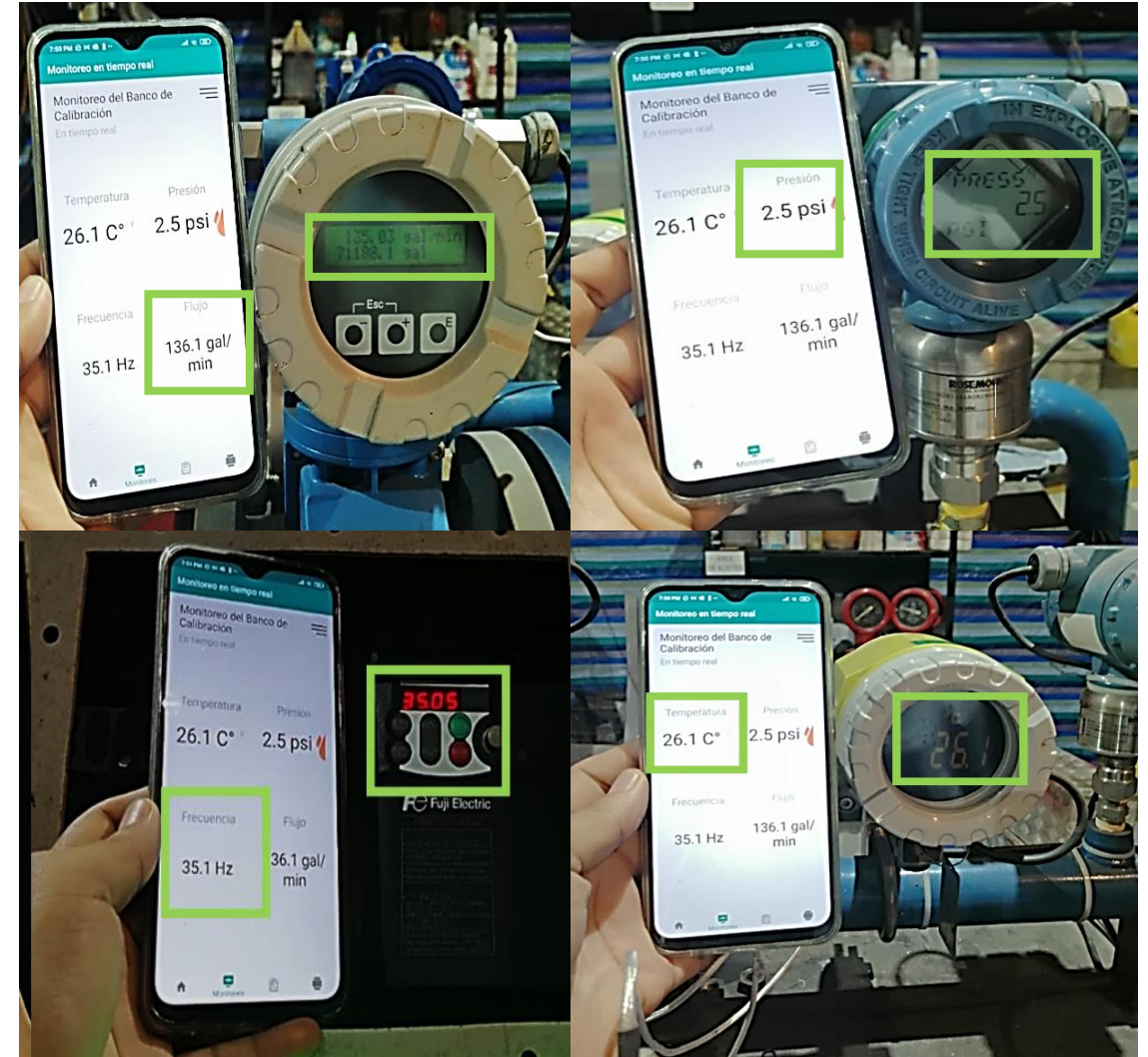
Al ingresar un usuario no registrado en la base de datos de Firebase se emite un mensaje de error, en cambio si se ingresa con un correo y contraseña correcta entrará a la venta de Menú de la aplicación



PRUEBAS Y RESULTADOS

Monitoreo en Tiempo real

Los valores de los sensores del banco de pruebas se comparan con los datos mostrados en la aplicación móvil.



PRUEBAS Y RESULTADOS

Datos primarios

Los datos primarios son los valores de los sensores que son registrados durante una calibración en el banco de pruebas.

DATOS PRIMARIOS PARA BANCO DE PRUEBAS DE CAUDAL				Identificación: FTM-033 Revisión: 002 Aprobación: 2021-07-16 Página: 1 de 1			
SERVICIOS DE INSTRUMENTACIÓN ESRAW S.A.							
1.- DATOS DEL SOLICITANTE:							
CLIENTE:	ESRAW S.A	TÉCNICOS:	JOSE RODAS	FECHA RECEPCIÓN:			
RIG:		LUGAR CALIB:	TALLER	DESCRIPCIÓN ADC:	VALIDACIÓN		
OPERADORA:		UBICACIÓN IBP:	BODEGA	PROCEDIMIENTO:	PTM-007		
DIRECCIÓN:		LOCACIÓN:		CÓDIGO ÚNICO:	ASC-0277-ATE		
2.- DATOS TÉCNICOS DE EQUIPOS:							
La evaluación del factor de riesgo en la trazabilidad con la relación de exactitud (TAR), de recomendaciones internacionales el patrón de trabajo debe ser 4 veces más exacto que el IBP.							
EQUIPO PATRÓN PRESIÓN			INSTRUMENTO BAJO PRUEBA				
Fabricante:	ROSEMOUNT CORP		Fabricante:	STAUFF			
Modelo:	VRS 100		Modelo:	SP6-BD5000			
Rango:	300 gal/min		No. Serie:	SNP2513100-A			
Exactitud:	0,5 %		Resolución:	-			
Resolución:	-		Exactitud:	2 %			
No. Serie:	E3-1841498		Rango Presión:	-			
Trazabilidad:	LME L 20215 DER		Rango Eléctrico:	-			
Incertidumbre:	0,9 gal/min		Coefficiente Temp:	-			
Fecha Calibra:	2021-09-15		Datos Extras:	-			
3.- DATOS AMBIENTALES & OTROS:							
Temp. In:	28,4	Temp. Out:	27,8	Tipo Fluido:	AGUA		
R.H. In:	52,2	R.H. Out:	51,3	Altitud:	258		
P. Al. In:	979,5	P. Al. Out:	979,5	Δ Altura:	-		
Inicio Calibración:	2022-02-03	Final Calibración:	2022-02-03	Próxima Calibración:	-		
4.- PROCEDIMIENTO PARA TOMA DE DATOS:							
La calibración se realiza por medición del banco de pruebas para turbinas, aplicando el procedimiento antes mencionado, se debe cumplir los siguientes detalles: Puntos de calibración según su exactitud: (2) dos ciclos ((Exact: > 0.1 % - 5 Puntos); 1 ciclo: (1 Ascenso & 1 Descenso).							
5.- PROCESO Y TOMA DE DATOS:							
Lecturas No.	Equipo Patrón Eswaw Eq. Frecuencia Hz	I.B.P. 1er Ciclo - ASCENSO			I.B.P. 1er Ciclo - DESCENSO		
		Presión psi	Temperatura °C	Caudal (gal/min)	Presión psi	Temperatura °C	Caudal (gal/min)
1	0	2,5	24,0	0,00	2,5	23,9	0,00
2	20,01	3,0	23,9	97,120	3,0	23,9	98,520
3	30,00	3,0	23,9	149,782	3,0	23,9	149,821
4	40,03	3,0	23,9	199,113	3,0	23,9	200,703
5	50,07	3,0	23,9	248,238	3,0	23,9	249,121
6	-	-	-	-	-	-	-
7	-	-	-	-	-	-	-
8	-	-	-	-	-	-	-
9	-	-	-	-	-	-	-
10	-	-	-	-	-	-	-
11	-	-	-	-	-	-	-

DATOS PRIMARIOS PARA BANCO DE PRUEBAS DE CAUDAL				Identificación: FTM-033 Revisión: 002 Aprobación: 2021-07-16 Página: 1 de 1			
SERVICIOS DE INSTRUMENTACIÓN ESRAW S.A.							
1.- DATOS DEL SOLICITANTE:							
CLIENTE:	ESRAW SA	TÉCNICOS:	W. Hidalgo	FECHA RECEPCIÓN:			
RIG:		LUGAR CALIB:	Taller	DESCRIPCIÓN ADC:	Validación		
OPERADORA:		UBICACIÓN IBP:	Bodega	PROCEDIMIENTO:	PTM-007		
DIRECCIÓN:		LOCACIÓN:		CÓDIGO ÚNICO:	ASC-0280-ATE-ESRAW		
2.- DATOS TÉCNICOS DE EQUIPOS:							
La evaluación del factor de riesgo en la trazabilidad con la relación de exactitud (TAR), de recomendaciones internacionales el patrón de trabajo debe ser 4 veces más exacto que el IBP.							
EQUIPO PATRÓN PRESIÓN			INSTRUMENTO BAJO PRUEBA				
Fabricante:	ROSEMOUNT CORP		Fabricante:	STAUFF			
Modelo:	VRS-100		Modelo:	SP6-B03000			
Rango:	300 gal/min		No. Serie:	SNP2813009-A			
Exactitud:	0,5 %		Resolución:	-			
Resolución:	-		Exactitud:	2 %			
No. Serie:	E3-1841498		Rango Presión:	-			
Trazabilidad:	LME L 20215 DER		Rango Eléctrico:	-			
Incertidumbre:	0,9 gal/min		Coefficiente Temp:	-			
Fecha Calibra:	2021-09-15		Datos Extras:	-			
3.- DATOS AMBIENTALES & OTROS:							
Temp. In:	27,2	Temp. Out:	28,0	Tipo Fluido:	Agua		
R.H. In:	48,7	R.H. Out:	50,2	Altitud:	258		
P. Al. In:	980,1	P. Al. Out:	980,1	Δ Altura:	-		
Inicio Calibración:	2022-02-04	Final Calibración:	2022-02-04	Próxima Calibración:	-		
4.- PROCEDIMIENTO PARA TOMA DE DATOS:							
La calibración se realiza por medición del banco de pruebas para turbinas, aplicando el procedimiento antes mencionado, se debe cumplir los siguientes detalles: Puntos de calibración según su exactitud: (2) dos ciclos ((Exact: > 0.1 % - 5 Puntos); 1 ciclo: (1 Ascenso & 1 Descenso).							
5.- PROCESO Y TOMA DE DATOS:							
Lecturas No.	Equipo Patrón Eswaw Eq. Frecuencia Hz	I.B.P. 1er Ciclo - ASCENSO			I.B.P. 1er Ciclo - DESCENSO		
		Presión psi	Temperatura °C	Caudal (gal/min)	Presión psi	Temperatura °C	Caudal (gal/min)
1	0	2,5	24,0	0,00	2,5	23,9	0,00
2	19,02	3,0	23,9	74,928	3,0	23,9	75,138
3	20,04	3,0	23,9	100,921	3,0	23,9	101,098
4	25,03	3,0	23,9	126,104	3,0	23,9	126,130
5	30,03	3,0	23,9	151,020	3,0	23,9	151,200
6	-	-	-	-	-	-	-
7	-	-	-	-	-	-	-
8	-	-	-	-	-	-	-
9	-	-	-	-	-	-	-
10	-	-	-	-	-	-	-
11	-	-	-	-	-	-	-

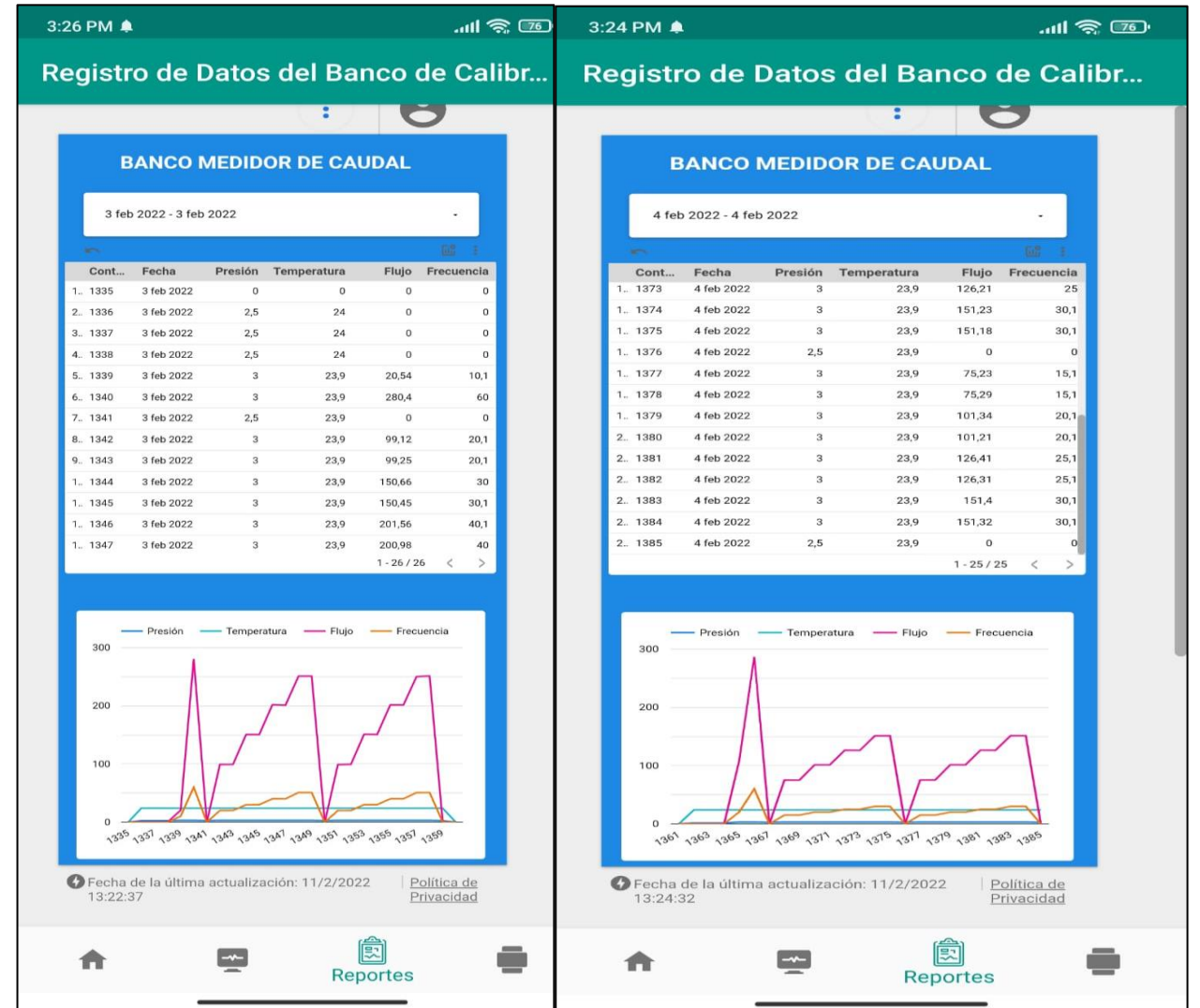


ESPE
UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

PRUEBAS Y RESULTADOS

Registros de la app

Se filtra por fecha los datos y la hora en la que se realizo la toma de datos primarios



Comparación de datos primarios vs datos del registro

de la app

En la tabla se presenta la comparación de los valores del sensor de caudal dado que es el que mayor variación presentó.

Pruebas	Datos Primarios	App IoT	Técnico
1	0	0	J. Rodas
2	97,12	99,25	J. Rodas
3	149,782	150,66	J. Rodas
4	199,113	201,56	J. Rodas
5	248,238	250,8	J. Rodas
6	0	0	J. Rodas
7	98,52	99,54	J. Rodas
8	149,821	151,15	J. Rodas
9	200,703	201,45	J. Rodas
10	249,121	251,14	J. Rodas
11	0	0	W. Hidalgo
12	74,928	75,21	W. Hidalgo
13	100,921	101,2	W. Hidalgo
14	126,107	126,3	W. Hidalgo
15	151,02	151,23	W. Hidalgo
16	0	0	W. Hidalgo
17	75,138	75,29	W. Hidalgo
18	101,098	101,34	W. Hidalgo
19	126,13	126,41	W. Hidalgo
20	0	0	J. Rodas

Nota. Valores expresados en gal/min



Monitoreo en Tiempo real

Los valores de los sensores del banco de pruebas se comparan con los datos mostrados en la aplicación móvil.



Pruebas de notificaciones del sistema



PRUEBAS Y RESULTADOS

Pruebas de reconocimiento del estado del sensor con AR



Cálculos T-Student

- Nivel de significancia (α): 0.05
- Media de mediciones de datos primarios (\bar{x}_1): 114.9 gal/min
- Media de mediciones App IoT (\bar{x}_2): 115.7 gal/min
- Desviación estándar de mediciones de datos primarios (s_1): 77.0
- Desviación estándar de mediciones App IoT (s_2): 77.6
- Tamaño de mediciones de datos primarios (n_1): 20
- Tamaño de mediciones de App IoT (n_2): 20



VALIDACIÓN DE LA HIPÓTESIS

Cálculos de T-Student

Estadístico t

$$t = \frac{(\tilde{x}_1 - \tilde{x}_2)}{\sqrt{\left(\frac{s_1^2}{n_1}\right) + \left(\frac{s_2^2}{n_2}\right)}}$$
$$t = \frac{(114.9 - 115.7)}{\sqrt{\left(\frac{77.0^2}{20}\right) + \left(\frac{77.6^2}{20}\right)}}$$
$$t = -0.03$$

Grados de libertad

$$gl = n_1 + n_2 - 2$$
$$gl = 20 + 20 - 2$$
$$gl = 38$$

Comparación de estadístico t

$$t \geq t_{\alpha/2}$$
$$0.03 \leq 1.686$$

Por lo tanto, aceptamos la hipótesis nula, la cual pronuncia que la implementación del Sistema Internet de las Cosas con realidad aumentada permitirá la digitalización, monitoreo y generación de eventos de un banco de calibración de medidores de caudal de la empresa ESRW S.A.



CONCLUSIONES Y RECOMENDACIONES

Conclusiones

- El proyecto se finalizó con el diseño e implementación del sistema IoT con realidad aumentada para la empresa ESRAW S.A. ubicada en la ciudad de El Coca, en la provincia de Orellana, la comunicación es vía WiFi con acceso a internet, es independiente de un computador, mediante el análisis de resultados se concluyó que el sistema IoT monitoriza el banco de pruebas de caudal en tiempo real.
- La tecnología IoT proporciona información en tiempo real de los valores de los sensores del banco de pruebas de caudal, desde cualquier parte del mundo que posea conectividad a Internet.
- Se implementó realidad aumentada importando una librería creada en el motor de videojuegos Unity, la cual accede a los datos de la base de datos mediante comunicación enviada desde la aplicación Android a un complemento de Unity creado con código C#, esta librería importada muestra el estado de los sensores del banco de pruebas de caudal con la utilización de la cámara del dispositivo android, enfocando a los marcadores designados para cada sensor.
- La implementación económica de este proyecto fue accesible, debido a que todos los softwares del sistema se ocuparon con licencia de acceso libre, el valor implementar los componentes electrónicos con una base elaborada por impresión 3D es de un valor menor a los 200 dólares.



CONCLUSIONES Y RECOMENDACIONES

Recomendaciones

- El dispositivo tecnológico en el que se instale la aplicación se recomienda deba tener una cámara posterior de 8 MP, un sistema android con versión 5 en adelante y una capacidad de memoria libre 200 MB para garantizar una experiencia de usuario fluida y la plena funcionalidad de la aplicación.
- Actualizar una vez cada seis meses la librería de Arduino FirebaseHttpClient dado que Firebase y Google actualizan sus direcciones 1 vez al año, y el fingerprint se modifica en la librería.
- Utilizar AR en aplicaciones industriales para optimizar los procesos al proporcionar información en tiempo real directamente a los operadores tiene un impacto positivo en la eficiencia y la toma de decisiones.
- La adopción de las tecnologías de este proyecto en aplicaciones industriales representa un paso audaz hacia la transformación digital y puede allanar el camino para una mayor excelencia operativa en un mundo empresarial en constante evolución.



GRACIAS POR SU ATENCIÓN



ESPE
UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA