UNIVERSIDAD DE TALCA

FACULTAD DE INGENIERÍA

ESCUELA DE INGENIERÍA EN MECATRÓNICA

SISTEMA DE MONITOREO DE VARIABLES DE PROCESOS DE EMPRESAS A TRAVES DE APLICACIÓN ANDROID.

Memoria para optar al Título de

Ingeniero en Mecatrónica.

Profesor Guía:

.

MICHAEL CARLOS VÁSQUEZ VÁSQUEZ

CURICÓ-CHILE

2020

CONSTANCIA

La Dirección del Sistema de Bibliotecas a través de su encargado Biblioteca Campus Curicó certifica que el autor del siguiente trabajo de titulación ha firmado su autorización para la reproducción en forma total o parcial e ilimitada del mismo.

Curicó, 2022

SISTEMA DE MONITOREO DE VARIABLES DE PROCESOS DE EMPRESAS A TRAVES DE APLICACIÓN ANDROID.

Michael Carlos Vásquez Vásquez

Junio 2020

Resumen

SISTEMA DE MONITOREO DE VARIABLES DE PROCESOS DE EMPRESAS A TRAVES DE APLICACIÓN ANDROID.

En esta memoria de título se tuvo como fin el diseño de un sistema de monitoreo de procesos a través de una aplicación para smartphone con sistema operativo Android.

Se presenta la descripción del funcionamiento del sistema de monitoreo, la programación de los dispositivos de monitoreo, la construcción de bases de datos para el manejo de datos online, la programación de los scripts PHP para la conexión entre los dispositivos, la aplicación y las bases de datos, además de la programación de una aplicación beta, hecha a través de una plataforma online básica.

Se verifica el funcionamiento correcto de los dispositivos, además del funcionamiento óptimo de la aplicación. Finalmente se presentan los anexos con todos los archivos utilizados para la programación.

Dedicado a mis padres, hermanos y novia.

Agradecimientos

La realización de este trabajo representa el final de mi etapa de estudios en la Universidad De Talca, este proceso fue bastante extenso, pero gracias a ciertas personas se pudo culminar.

Tabla de Contenidos

Lista de Tablas

Lista de Figuras

Capítulo 1. Introducción

1.1. Introducción

Esta memoria de título contempla el diseño de un sistema de monitoreo en líneas de producción y procesos en empresas, el cual se basa en dos tipos de dispositivos, el primero es un sensor del tipo contador, que permitirá contar la cantidad de productos que se produzcan en una línea de embazado o embotellado, y el segundo dispositivo permitirá medir las variables de sensores Siemens, los cuales poseen salidas analógicas que van de los 4 a 20mA. Los dispositivos mencionados anterior mente se conectarán mediante wifi a una base de datos online, la cual estará conectada a una aplicación Android, la cual obtendrá y procesará estos datos para ser visualizados en tablas y gráficos, además de programar alarmas de correo electrónico, esto solo si alguna de las variables sensadas sobrepasa un valor preestablecido previamente.

1.2. Estado del Arte.

En cuanto a sistemas de monitoreo a través de aplicaciones móviles hay que empezar con algunos datos sobre estos.

Al día de hoy los dispositivos móviles son aptos para incorporar casi cualquier componente, sea de software o de hardware, para mejorar o ampliar su función inicial. Siendo los más frecuentes la conexión a internet y la telefónica. (Garita-Araya, 2013)

En muy pocos años los teléfonos móviles han llegado a convertirse en un dispositivo esencial para particulares o empresas con posibilidad de elección entre miles de modelos que los diferentes fabricantes siguen innovando intentando destacar sobre los de la competencia. (Lucas)

Teniendo los dos puntos anteriores en cuenta, el uso de estos dispositivos para el monitoreo de procesos es lo más viable, puesto que, estos son cada vez más frecuentes y poseen todas las herramientas de software necesarias para cumplir esta función.

A continuación, algunos datos estadísticos a tener en cuenta sobre los smartphones.

- El 68% de la población mundial ya cuenta con móvil, mientras que internet tiene una penetración tan solo del 53%. (Ditrendia, 2019)
- En 2017, no solo ha crecido el tiempo que dedicamos a los dispositivos móviles, sino que el número de usuarios que se declara ‘solo móvil’ ha aumentado en casi todas las regiones. (Ditrendia, 2019)
- En el mundo, el 52% del total del tráfico web se realiza desde el móvil (4% más que en 2016). (Ditrendia, 2019)
- El internet de las cosas (IoT-Internet of Things) será la tecnología con mayor crecimiento de aquí a 2020, siendo los millennials el rango de edad que más apuesta por esta tecnología. (Ditrendia, 2019)
- Las Apps suponen ya más del 80% del tiempo que dedicamos al uso del móvil. (Ditrendia, 2019)
- En 2017 se descargaron 178,1 millones de Apps en el mundo. (Ditrendia, 2019)

- Un smartphone tiene de media 80 aplicaciones instaladas, de las cuales solo se usan la mitad. (Ditrendia, 2019)

Según los datos mencionados anteriormente, la utilización de smartphone, aplicación y el IoT, son una constante en la actualidad, prácticamente más de la mitad de la población mundial cuenta con un móvil, además de que cada vez somos más dependientes de estos y de sus aplicaciones.

Por otro lado, en chile, "un 72% afirmó que el smartphones es la herramienta que más usan en sus trabajos. De ellos, el 81% de las personas en el país consideraron que pueden aumentar su productividad si sus celulares fueran más potentes y pudieran reemplazar a otros dispositivos como el computador" (Vilches, 2019).

Los datos anteriores confirman que, en Chile, los trabajadores están ligando cada vez sus trabajos a sus smartphones, y que la idea de tener aplicaciones específicas para sus trabajos es la nueva tendencia en estos tiempos. Con la salida de celulares más potentes, las capacidades de las aplicaciones serán idóneas para realizar cualquier tipo de trabajo, por ende, el monitoreo de procesos será de forma más fluida y exacta.

También es necesario tener una definición de las aplicaciones.

Una aplicación (también llamada app) es simplemente un programa informático creado para llevar a cabo o facilitar una tarea en un dispositivo informático. Cabe destacar que, aunque todas las aplicaciones son programas, no todos los programas son aplicaciones. Existe multitud de software en el mercado, pero sólo se denomina así a aquel que ha sido creado con un fin determinado, para realizar tareas concretas. No se consideraría una aplicación, por ejemplo, un sistema operativo, ni una suite, pues su propósito es general. (sistemas.com, 2019)

Hoy en día la cantidad de aplicaciones y la cantidad de tiempo invertido en ellas es abismante, por ende, el uso de aplicaciones móviles para mejorar o supervisar procesos en las empresas, el conocimiento es cada vez más importante en las empresas y se necesita un sistema amigable para las personas que no conocen de forma adecuada los procesos, pero necesitan tomar decisiones.

En este momento, las aplicaciones para smartphone se encuentran en todos los lugares y dentro de todas las rutinas diarias de las personas, no es una exageración decir: “hay una aplicación para todo, y si no existe, pronto lo estará”, esto quiere decir que el ámbito del monitoreo de empresas no es la excepción. Gran parte de las aplicaciones que se encuentran disponibles en este momento, son de manejo de personal (horarios, sueldos, vacaciones, etc.), optimización de procesos financieros y de marketing, en el área del diseño y la construcción también existen aplicaciones que permiten visualizar el estado final de construcciones, como es el sistema SieteVisión, el cual permite superponer la imagen del diseño final sobre el lugar de la construcción, esto para prevenir y corregir errores de diseño antes de dar comienzo a las faenas.

Otro uso que se le ha dado a las aplicaciones, es la de conectarse a cámaras IP, con las cuales poder ver en tiempo real lo que sucede en diversos lugares, además de programar alarmas, que se activan a través de sensores de presencia.

Estos son algunos ejemplos de usos de aplicaciones para smartphone, pero en el ámbito de monitoreo de variables de procesos específicos de empresas, no se han encontrado aplicaciones que cumplan esta función. El único caso que se ha encontrado es un proyecto ganador de un fondo concursable Corfo, este proyecto fue llamado “Aquadata”, el cual consistía en un sistema de monitoreo de variables de procesos de plantas de tratamiento de aguas, estos procesos eran, nivel y flujo de agua, nivel bajo o alto de cloro y de decloro, sensor de presencia de burbujas, flujo de cloro y decloro. Este sistema consistía en un dispositivo basado en Raspberry Pi y una pantalla táctil, en la cual se refleja en una interfaz gráfica que permitía visualizar los sensores activos, esto también consistía en una interfaz web, en la cual también se reflejaran los sensores y se podrá programar alarmas.

Este sistema fue implementado en una planta de tratamiento de agua en la ciudad de Talca, por la empresa Ecoglobal por encargo de la constructora Ecofya, este sistema se probó satisfactoriamente, pero por falta de recursos se dejó de lado por la empresa Ecoglobal.

Lo que diferencia al sistema de este proyecto del de la empresa Ecoglobal, es la utilización de una aplicación única para dispositivos Smartphone, esto es un cambio fundamental en

el sistema, ya que, esto permite al usuario estar al pendiente de todos los procesos al instante y sin necesidad de ingresar a un navegador web.

Otra de los cambios es la modularización de los sensores y dispositivos, permitiendo instalar estos en lugares diversos sin necesidad de crear extensiones que conecten los sensores a un módulo principal, esto permite abaratar costos de instalación y evitar que el sistema falle por completo si uno de los dispositivos falla.

1.3. Hipótesis de Trabajo.

Utilizando los medios disponibles obtenidos a través de un proyecto ganador de los fondos Sercotec Semilla, es posible realizar la creación de un sistema de monitoreo de procesos a través de una aplicación Android. Este sistema consta del dispositivo a montar en el proceso, la base de datos en el servidor web y la aplicación Android.

1.4. Objetivos

1.4.1. Objetivo General

- Creación de un sistema de monitoreo de procesos en empresas, a través de una aplicación para smartphone con sistema operativo Android, con 2 dispositivos de monitoreo.

1.4.2. Objetivos Específicos

- Selección de componentes para dispositivos de monitoreo.
- Programación Dispositivo contador de productos y dispositivo lector de sensores.
- Creación bases de datos en servidor web.
- Programación de Script PHP en sitio web para consultas en base de datos.
- Creación aplicación Android para smartphone.

.

1.5. Alcances y Limitaciones

Tampoco consta de señal wifi propia, estos dispositivos se conectan a través de la señal wifi que proporciona la empresa en que se instalen. Además, la aplicación solo será para dispositivos con sistema operativo Android.

- El proyecto se limita al diseño de ambos dispositivos sensores y la prueba de funcionamientos de estos.
- Los dispositivos no constaran de señal wifi propia, estos dispositivos se conectarán a través de la señal wifi que proporciona el lugar en que se instalen.
- La aplicación solo será para smartphone con sistema operativo Android.

- No incluye la prueba de estos dispositivos en empresas con líneas de producción o embazado.

1.6. Temario

En el presente documento se da a conocer el desarrollo del sistema, en donde se resalta la programación de los dispositivos de monitoreo y la aplicación, con el ensamblaje de los dispositivos, además de la creación de las bases de datos y la programación de los scripts para el acceso a ellas. En el capítulo 2 se identificar los componentes a utilizar, se desarrollarán los códigos de programación, se definirá la estructura de los dispositivos de monitoreo. En el capítulo 3 se mostrarán los resultados del funcionamiento del sistema de monitoreo. Para finalmente en el capítulo 4 se darán las conclusiones y los anexos que incluyen los programas utilizados.

Capítulo 2. Diseño sistema de monitoreo

2.1. Descripción del sistema de monitoreo de procesos

El sistema de monitoreo consta de 4 puntos claves, el primero son los dispositivos de monitoreo que tomaran los datos de los procesos, luego estos serán enviados, por medio de la conexión wifi del dispositivo, hacia los scripts PHP que almacenaran lo enviado en las bases de datos. Lo segundo corresponde a los scripts alojados en la página web, los cuales como se explicó anteriormente, almacenaran los datos, además de realizar las consultas realizados a través de la aplicación Android, aparte de realizar algunas funciones para simplificar la programación de la app. El tercer punto son las bases de datos que almacenaran todos los datos, sean datos de los usuarios tanto como los datos enviados por los dispositivos de monitoreo. El cuarto y último punto es la aplicación Android que permitirá al cliente revisar todos los datos de sus procesos.

Con respecto a los dispositivos, se construirán dos dispositivos, el primero está orientado a las empresas con líneas de producción dígase empresas de empaquetado o embotellado, el cual permitirá contar con precisión la cantidad de productos que pasen frente de él, como características principales de este dispositivo destacan:

- Conexión a red de 220v.
- Conexión wifi.
- Reinicio a 0 del contador.
- Conteo de producto con intervalos que van desde los 0.5 segundos.
- Led de indicación de encendido.
- Interruptor de encendido y apagado.
- Tamaño compacto, no invasivo.

El segundo dispositivo está orientado a los procesos diversos de las empresas, en los que sea imprescindible obtener información fiable y detallada de las distintas variables de esos procesos, como pueden ser temperatura, caudal, nivel, presión, etc., en estas empresas estas variables están siendo monitoreadas ya por sensores siemens, pero esta información no es de fácil acceso y generalmente es usada solo para controlar esos procesos mediante un ciclo cerrado, para aumentar el conocimiento de esas variables se utiliza este dispositivo que consta de las siguiente características.

- Conexión a red de 220v.
- Conexión wifi.
- Ajuste de tiempo de sensado de variables.
- 2 entradas de sensores con salida analógica de 4 a 20 mA.
- Led de indicación de encendido.
- Interruptor de encendido y apagado.
- Tamaño compacto, no invasivo.

Cabe mencionar que ambos dispositivos envían sus datos a una base de datos online la cual se conecta a la aplicación Android, esta aplicación conta de las siguientes características:

- Compatibilidad con dispositivos con sistema operativo Android.
- Logeo de usuario.
- Tabla de sensores disponibles.
- Sistema de alarmas vía email.
- Tabla de datos de cada sensor o dispositivo.
- Gráficos de los datos de cada sensor o dispositivo.
- Cambio de contraseña.

La comunicación de los dispositivos con las bases de datos y la aplicación Android, es como se detalla en el esquema que se muestra a continuación.

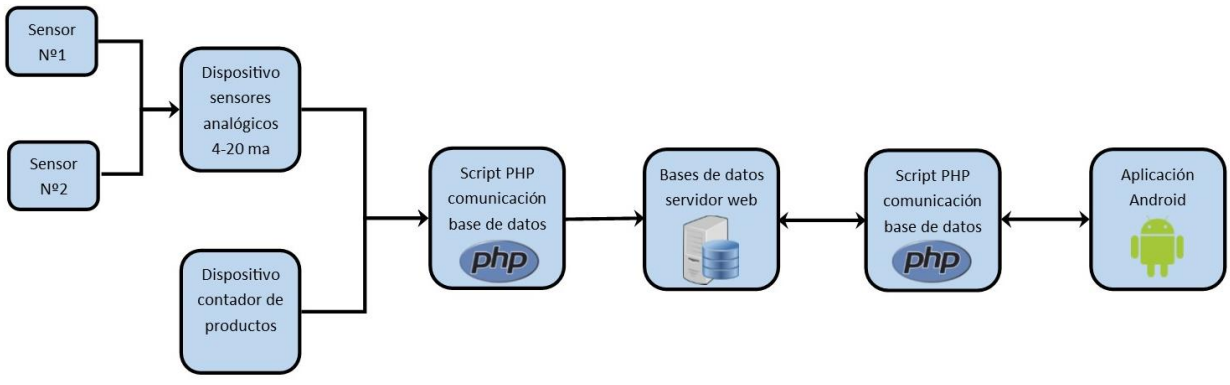


Fig. 2.1 Esquema comunicación dispositivos de monitoreo.

Como se puede observar en el esquema anterior la comunicación entre los dispositivos y las bases de datos es en un solo sentido, los dispositivos solo envían datos hacia un script en PHP que se encarga de hacer las solicitudes a la base de datos, esto es porque son dispositivos de monitoreo solamente, o sea, no influyen en los procesos que están sensando. No obstante, la comunicación entre las bases de datos y la aplicación de smartphone tiene que ser de doble vía, enviar y recibir datos de la base de datos, esto es necesario para poder programar alarmas, efectuar el cambio de contraseñas y realizar diversas solicitudes en las bases de datos.

2.2. Estructura del trabajo

Para un mejor análisis del sistema de monitoreo este se dividirá en 7 partes, las cuales serán enlistadas a continuación:

- Selección de componentes.
- Diseño de la estructura de dispositivos.
- Programación dispositivo contador.
- Programación dispositivo sensores siemens.
- Programación acceso a base de datos.
- Programación aplicación Android.

2.2.1. Selección de componentes para los dispositivos de sensado.

La selección de componentes se realizará en 2 partes, la primera serán los componentes del dispositivo de conteo de líneas de producción y la segunda serán los componentes del dispositivo de lectura de sensores siemens, también cabe destacar que las especificaciones técnicas de los componentes de los dispositivos, fueron sacadas de las páginas web de los proveedores de estos, estas se encuentran en la bibliografía de este documento.

2.2.2. Componentes del dispositivo de conteo.

Para iniciar la elección de componentes, se empezará por la placa de control, esta permitirá al dispositivo recibir el conteo del número de botellas y además enviar estos datos a la base de datos, por ende, uno de los factores a tomar en cuenta al momento de

seleccionarla es que tenga una conexión wifi integrada y que sea de tamaño compacto, esto para que el dispositivo no alcance dimensiones muy exageradas. También la placa controladora deberá tener una buena capacidad de almacenamiento de datos y además de tener una relativamente buena velocidad de envió de datos mediante el wifi.

Se selecciono a la Raspberry Pi Zero W, para controlar este dispositivo, ya que, cuenta con todas las características que necesita este, constando con wifi incorporado, un pequeño tamaño y unas capacidades realmente aceptables.

Fig. 2.2 Imagen referencia Raspberry pi Zero W.

Una vez seleccionada la placa de control se procede a la selección de los periféricos necesarios para realizar el conteo, para esto es necesario un sensor que mida interrupciones, para esto se utilizar un sensor Grove de proximidad infrarrojo con un alcance máximo de 80 cm, ya que, este medirá de manera más confiable las interrupciones ocasionados por los productos al ser contados.

Fig. 2.3 Imagen sensor de proximidad infrarrojo Grove.

Uno de los problemas de usar la Raspberry Pi es que no posee entradas analógicas, para solucionar este inconveniente se utilizará el módulo conversor análogo digital ADS1115, que permitirá agregar 4 entradas analógicas con una resolución de 16 bits.

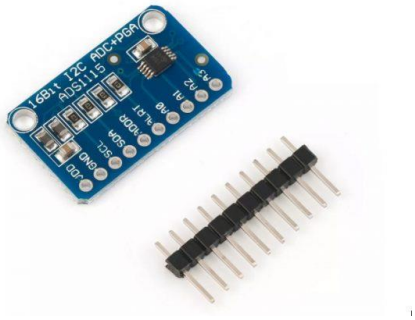


Fig. 2.4 Imagen modulo conversor análogo digital Ads1115.

Para las demás funciones del dispositivo se utilizarán un pulsador normal que se utilizará para reiniciar el contador a 0 y se utilizar un led verde de 5mm para identificar en qué momento esta encendido el dispositivo.

Fig. 2.5 Imagen referencia pulsador de reseteo.

Fig. 2.6 Imagen referencia led de encendido.

Por último, para la alimentación del dispositivo se utilizará la fuente de alimentación de la Raspberry Pi, esta es un cargador de 5v con 3A, con esto es suficiente para la alimentación de todos los componentes.

Fig. 2.7 Imagen referencia fuente de energía dispositivo.

A continuación, se muestra el esquema de conexionado de los componentes fundamentales para el sensado del dispositivo de conteo.

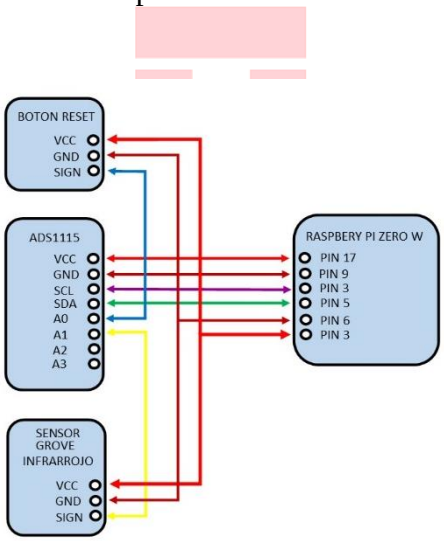


Fig. 2.8 Esquema conexionado sensores del dispositivo de conteo.

Como se puede apreciar en el esquema anterior, las conexiones más importantes para este dispositivo, son la conexión al módulo conversor análogo-digital ADS1115, el cual permitirá leer los valores analógicos que envían el sensor infrarrojo y el botón de reset, aunque el botón de reset es un dispositivo con señal digital, se usara a través del módulo conversor, para simplificar el script en Python de la Raspberry pi, y se tomara como una variable analógica. Cabe destacar que la alimentación de los módulos sensor y del

conversor A/D es de 3,3 Vdc, puesto que, la Raspberry Pi Zero W sus señales digitales trabajan a ese voltaje.

2.2.3. Componentes del dispositivo de sensores Siemens.

Este dispositivo permitirá leer las señales analógicas dadas por los sensores Siemens, la mayoría de los sensores siemens usados en las empresas poseen una salida analógica que va de 4 a 20 mA, este dato será tomado cada cierta cantidad de tiempo, el cual va desde los 10 segundos hasta los 15 minutos, también, estos datos serán enviados mediante wifi hacia la base de datos, por ende, estas serían las cualidades de este dispositivo.

Para el control del dispositivo se ha optado por utilizar una placa Nodemcu v3, la cual ya viene con conexión wifi, también se ha optado por esta placa, ya que, es de bajo costo y de fácil programación, además de que por el uso del dispositivo no necesita una gran velocidad de envío de datos.

Fig. 2.9 Imagen referencia tarjeta Nodemcu.

Para ajustar el tipo en que el dispositivo leerá los datos del sensor Siemens, se utilizará un Potenciómetro (Trimpot) de 10K.

Fig. 2.10 Imagen referencia Trimpot de 10K.

Para la lectura de los datos que vienen de 4 a 20 mA, se utilizarán dos módulos conversor corriente a voltaje, que permitirán convertir esa señal a una de 0 a 5V, la cual podrá leer la placa.

Fig. 2.11 Imagen referencia modulo conversor corriente a voltaje.

En este dispositivo se utilizarán varios componentes que, en el dispositivo anterior, los cuales son el conversor análogo digital ADS1115, ya que, el Nodemcu solo posee una entrada analógica, también se utilizaran el led y la fuente de alimentación, puesto que, el Nodemcu funciona al mismo voltaje que la Raspberry Pi.

A continuación, se muestra el esquema del conexionado interno, del dispositivo de monitoreo de sensores externos análogos.

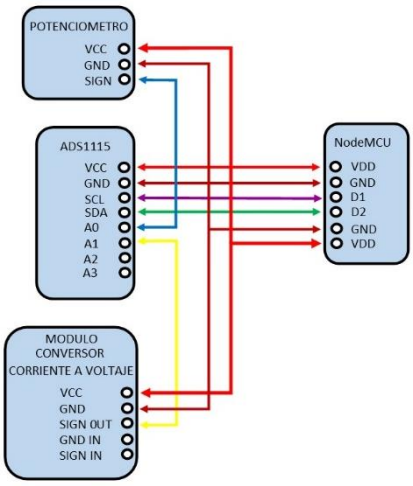


Fig. 2.12 Esquema conexionado componentes del dispositivo de sensores.

Como se puede ver en el esquema anterior, los componentes fundamentales del dispositivo de monitoreo, los cuales son, la placa Nodemcu, el conversor análogo-digital y el potenciómetro. Es este esquema se muestra un solo modulo conversor, pero en realidad son 2, el que no se muestra se conecta a la entrada analógica 2 del módulo análogo-digital.

2.3. Diseño estructura dispositivos de monitoreo.

La estructura de los dispositivos de monitoreo se realizó en base a los componentes a utilizar en ellos, tomando como base que todas las paredes del dispositivo tengan como mínimo 2 mm de espesor, ya que, estos dispositivos serán construidos con una impresora 3d, y al probar diferentes impresiones en esta, se dio cuenta de que el espesor mínimo para que una pieza quede en condiciones óptimas para su uso es de 2mm.

Cabe destacar que ambos dispositivos serán confeccionados con filamento PLA de 1.75mm, esto por la versatilidad de este material, además de que este ha producido mejores resultados al momento de imprimir, menos fallas de impresión, al contrario del ABS.

2.3.1. Diseño estructura dispositivo de conteo.

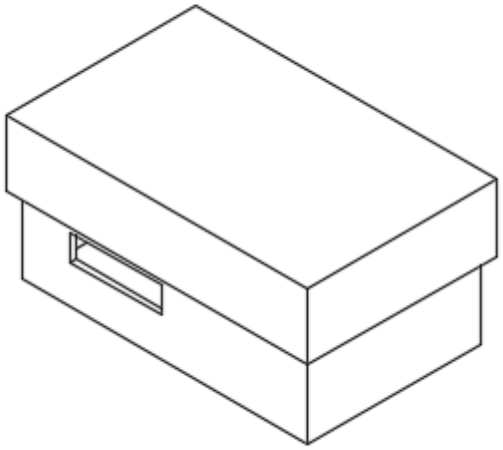


Fig. 2.13 Vista previa dispositivo de conteo.

En la imagen anterior se puede ver la apariencia del primer dispositivo, se puede ver una apertura rectangular en una de sus caras, en esta, va instalado el sensor infrarrojo, el cual permitirá realizar el conteo de los productos. Este dispositivo consta de una tapa que sellara al dispositivo.

2.3.2. Diseño estructura dispositivo monitoreo de sensores.

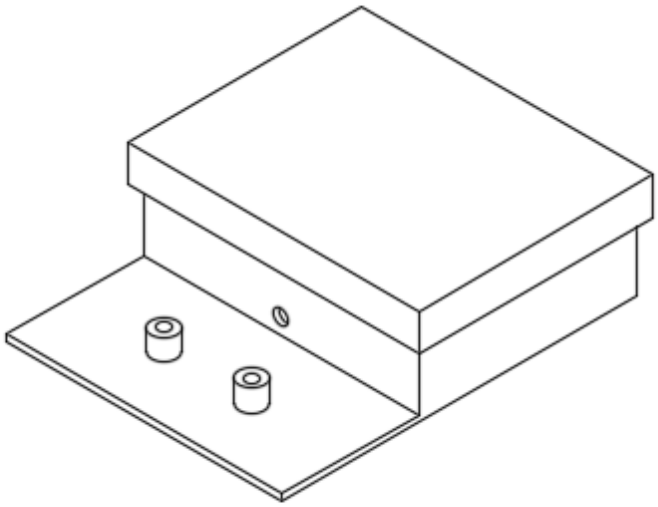


Fig. 2.14 Vista previa dispositivo de monitoreo de sensores.

Como se puede apreciar en la imagen anterior, el segundo dispositivo es un tanto distinto al primero, esto se debe a que este dispositivo se debe conectar directamente a los sensores

a medir, para lograr esto, evitando que se tenga que abrir el dispositivo, se creó una sección del dispositivo en la cual se adjuntara una regleta de conexionado de 4 entradas y salidas, las 4 entradas estarán conectadas directamente a los módulos conversores al interior del dispositivo y las salidas quedaran libres para conectar en este caso 2 sensores.

Los planos más detallados de ambos dispositivos se encuentran en los anexos de este documento, en el anexo k.

2.4. Base de datos sistema de monitoreo.

Las bases de datos corresponden a tablas online, en donde se pueden almacenar datos para posteriormente utilizarlos de en algún otro momento. Para este sistema este tipo de recursos son muy útiles para guardar información de los usuarios y datos de los sensores. Las bases de datos a utilizar son de tres tipos (registro de clientes, sensores empresas, datos de sensores).

2.4.1. Base de datos clientes.

La base de datos de los usuarios de la aplicación, consta de 7 campos los cuales son, id, nombre, apellido, empresa, mail, clave y sensores. Los campos anteriores son llenados con los datos de los clientes registrados, como se ven en la tabla siguiente.

Id	nombre	apellido	empresa	mail	clave	sensores
1	michael	vasquez	monitoreolocal	mcvv@gmail.com	super4	3
2	carlos	bustos	ebema	cb34@gmail.com	este23	4
3	ryan	gold	trescat	trgd@gmail.com	Sstr22	6

Tabla 2.1 Base de datos.

Como se puede ver en la tabla anterior, el id corresponde al número único de cada cliente, con este digito se puede identificar fácilmente al usuario para futuras consultas en las bases de datos. El campo sensores corresponde a la cantidad de sensores que tiene registrado esa persona. Por último, el campo clave corresponde a la clave única que se le

asigna al cliente para que pueda iniciar sección en la aplicación, cabe destacar que esta contraseña es asignada al cliente y no creada por él.

2.4.2. Base de datos sensores.

Las bases de datos de los sensores que poseen los clientes, contienen los siguientes datos: ID, TIPO, DESCRIPCION, RANGO, SETPOINT y ALARMA. Un ejemplo de estas bases de datos es la siguiente tabla.

ID	TIPO	DESCRIPCION	RANGO	SETPOINT	ALARMA
3	DIGITAL	Sensor de luz ambiental	0-1	900	off
2	ANALOGO	Sensor de temperatura ambiental	-5 – 50 ºC	-3	Off
1	CONTADOR	Contador de botellas	1 en 1	50	Off
4	ANALOGO	Sensor de humedad	0-100%	70	Off

Tabla 2.2 Base de datos sensores de cliente.

Como se puede apreciar en la tabla anterior, cada sensor posee un tipo, este tipo corresponde al funcionamiento en si del sensor, estos son de 3 tipos (ANALOGO, DIGITAL Y CONTADOR), el del tipo análogo permite tomar datos que varían durante el tiempo, por ejemplo, la temperatura de una sala. El del tipo digital, permite verificar el estado en de un proceso, dando una señal si se ha cumplido una condición determinada o no, como ejemplo se puede tomar si hay luz en una sala o no. El ultimo tipo es el contador, este permite realizar conteos de productos o entes.

El campo descripción corresponde a un pequeño texto que permite identificar que va a medir el sensor en si como, por ejemplo, un sensor de luz. El campo rango permite identificar entre qué puntos trabaja el sensor, como por ejemplo un sensor de temperatura tiene un rango entre -5 y 50ºC.

Por último, los campos set point y alarmas están relacionados, el set point es establecido a través de la aplicación y este permite disparar la alarma, si el valor del campo alarma está en “on”.

2.4.3. Base de datos valores de sensores.

Esta base de datos contiene todos los valores obtenidos a través de los sensores de todos los clientes, teniendo como campos (empresa, índice, valor, fecha y hora). A continuación, se muestra una tabla de ejemplo que permite ver el uso de los campos.

Empresa	Índice	Valor	Fecha	Hora
1	1	1	10/09/18	12:47:45.84
1	1	2	10/09/18	12:47:47.15
1	1	3	10/09/18	12:47:48.23
1	1	4	10/09/18	12:47:49.25
1	1	5	10/09/18	12:47:50.34

Tabla 2.3 Base de datos valores de sensores.

Como se puede apreciar en la tabla anterior, esta posee 5 campos, el primero corresponde a el número de identificación único de la empresa, en este caso es el número 1, el segundo es el índice, este es el id del sensor que envía el dato registrado, el campo valor guarda el dato registrado en ese instante por el sensor, en el caso del ejemplo corresponde al valor de un numero de un sensor de conteo, la fecha registra el día en que se tomó el dato y por último la hora corresponde al instante de la toma del dato del sensor.

2.5. Programación dispositivos de monitoreo.

La programación de ambos dispositivos es similar en funcionalidad, puesto que, ambos realizan una función equivalente, la cual es recibir datos y enviarlos a la página web, el diagrama de flujo de la programación de los dispositivos es el siguiente.

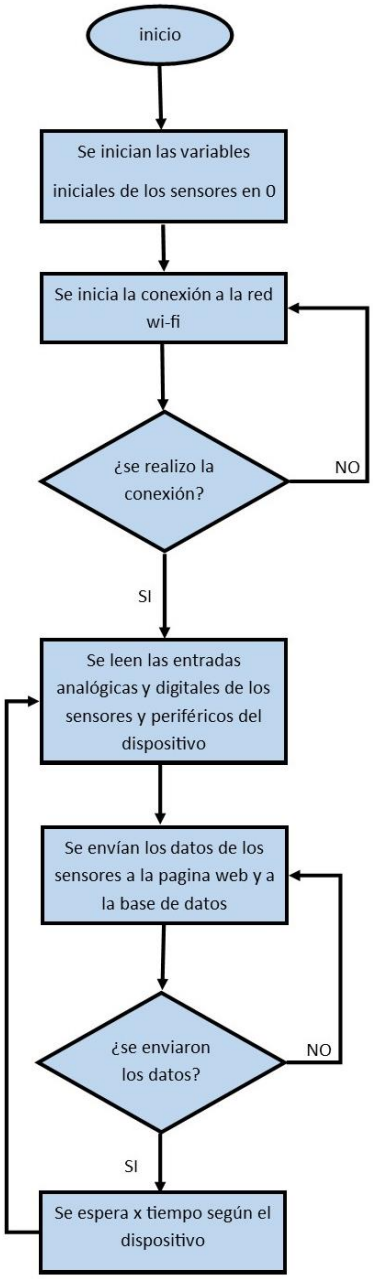


Fig. 2.15 Diagrama de flujo programación básica dispositivos de monitoreo.

Como se puede ver en el diagrama anterior, lo principal del programa es conectar el dispositivo a la red wifi, sin ese paso el dispositivo no podrá cumplir su función de enviar los datos a la página web. También posee un bucle que se reinicia mientras que el dispositivo este encendido, este bucle consta de la lectura de los sensores y su posterior envió a la base de datos.

2.5.1. Programación dispositivo contador.

El dispositivo contador utilizara como placa de control la Raspberry Pi Zero W, la cual se compone de la placa en sí y una memoria microSD de 16gb, con la cual se instala el sistema operativo de la Raspberry Pi, en este caso Raspbian, para la programación de este dispositivo se optó por el lenguaje Python, el cual es más fácil de comprender y hay bastante documentación de su uso en diversos aspectos.

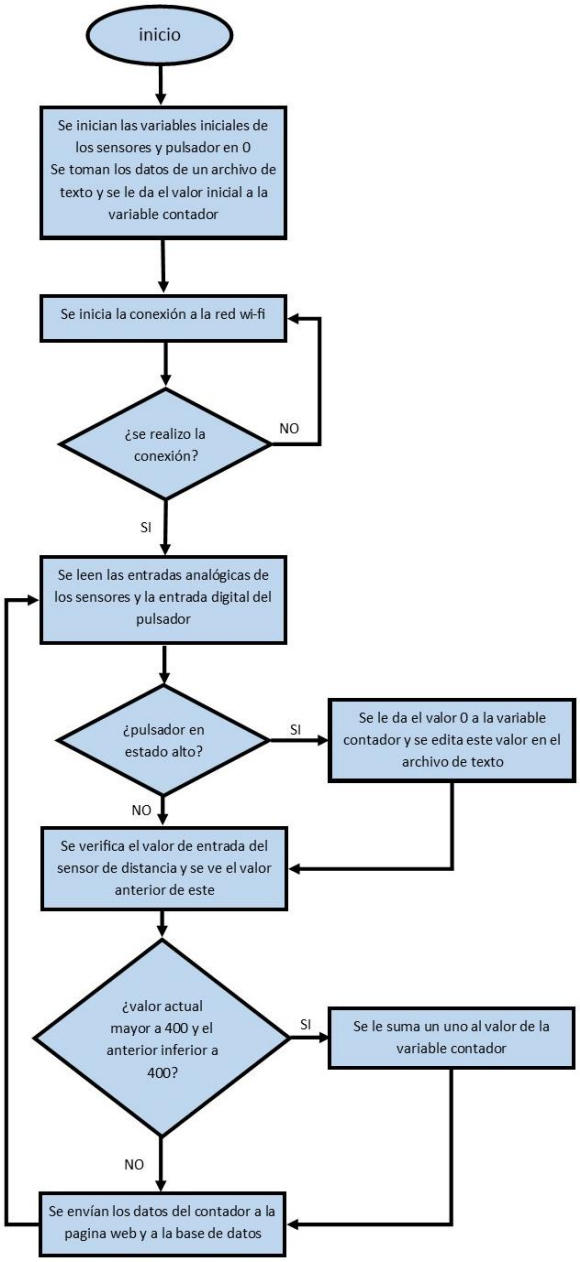


Fig. 2.16 Diagrama de flujo programación de dispositivo contador.

Como se puede apreciar en el esquema anterior, la programación del dispositivo consta de un ciclo que consta en tomar los datos recogidos por el sensor de proximidad, si este valor es mayor a 400 y el valor anterior a este es menor a 400, se le suma un uno a la variable contador y se guarda en un documento de texto, y si no, no se modifica este valor, este es el funcionamiento en sí, del sistema de conteo, este programa consta también de un reseteo a través de un pulsador, si este es pulsado, el sistema se reinicia a 0, guardándose este valor en el archivo TXT y en la variable contador.

2.5.2. Programación dispositivos de sensores Siemens.

El dispositivo de monitoreo de sensores periféricos con salida de lectura analógica que va desde 4 a 20 mA, tendrá 2 o más conexiones de entrada, con las cuales la placa controladora, en este caso una placa Nodemcu, como esta placa solo consta de 1 entrada analógica, se utiliza un conversor análogo digital ADS1115.

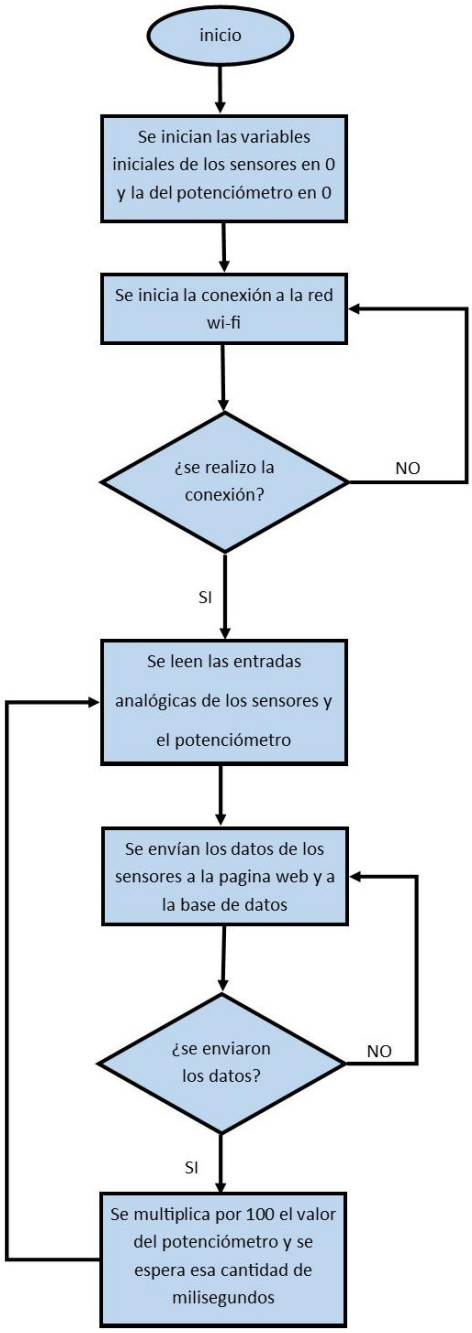


Fig. 2.17 Diagrama de flujo programación dispositivos sensores.

Como se puede notar en el esquema anterior, la programación de este tipo de dispositivo, es diferente al del de conteo, puesto que, esta es solo de toma de datos de sensores, el ciclo es leer los sensores y enviarlos mediante la conexión wifi, otra diferencia es que la toma de datos de los sensores es controlada mediante la señal analógica de un potenciómetro que, mediante el valor de esta, es el tiempo de toma de datos del dispositivo.

2.6. Programación acceso a base de datos.

La programación para el almacenamiento de los datos que provienen de los dispositivos de monitoreo, se hace en el lenguaje PHP a través de un script, este recibe los datos de estos por medio de peticiones básicas del tipo “INSERT”, como se puede ver en la imagen del script a continuación.

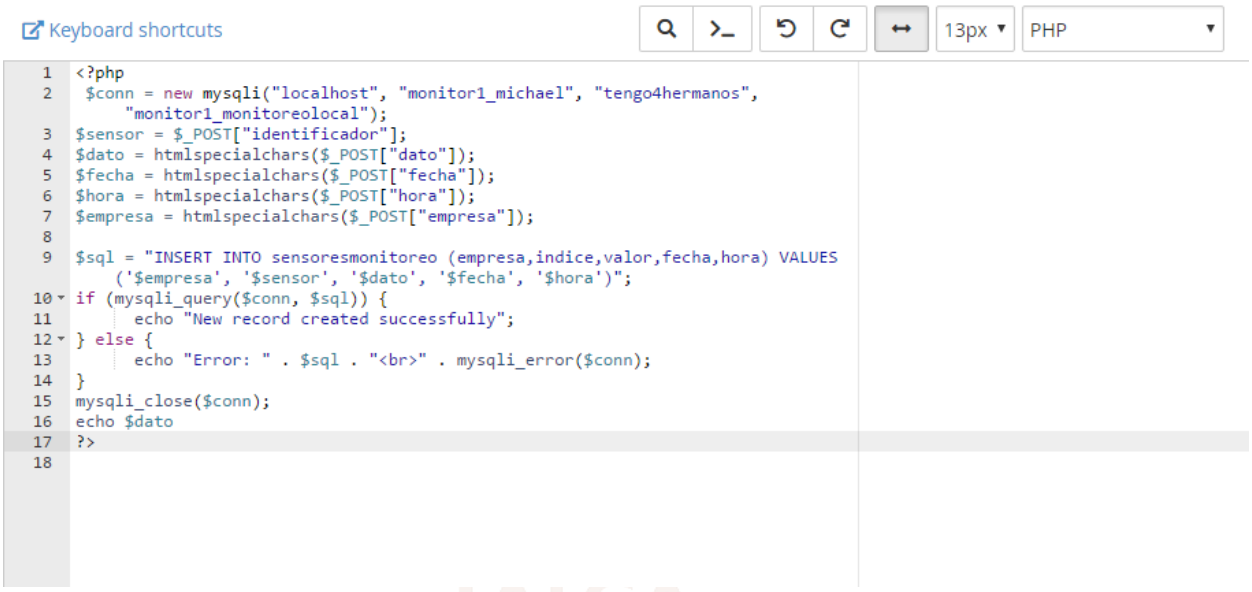


Fig. 2.18 Imagen programa almacenamiento de datos.

Como se puede ver en la imagen anterior el código la inserción de los datos es bastante simple y corta, con apenas 17 líneas de código, en donde primero se abre la conexión a la base de datos a través del nombre del host, el nombre de usuario, el nombre de la base de datos y la clave de usuario, esto se puede ver en la línea 2. Este script revise los datos por vía Post y los asigna a diversas variables, las cuales después son insertados en la base de datos a través de la línea 9, donde se selecciona la tabla de la base de datos, los campos que se insertaran y las variables a insertar.

La totalidad de los scripts utilizados se basan en consultas a la base de datos, en ellos los comandos básicos son reiterativos, estos son las consultas a las bases de datos, a continuación, se presentan ejemplos de estas líneas de código.

$$\$con = mysqli\_connect("localhost","genesist\_usermaster","genesisdual2020","genesist\_appdual"); \quad (2.1)$$

La línea de código anterior permite conectarse a la base de datos, en este caso la base de datos se llama “genesist_appdual”. Para conectarse se debe usar la función “mysqli_connect”, esta función necesita 4 campos para conectarse, que son: el host en que esta alojada la base de datos, el nombre del usuario y la clave de este mismo (el usuario y la clave se configuran en la aplicación Base de datos Mysql, que se encuentra en el cpanel del hosting), estos datos se ingresan en ese orden como se aprecia en el ejemplo anterior.

$$\$sql = "INSERT\ INTO\ sensoresmonitoreo\ (indice, valor, fecha)\ VALUES\ ('\$sensor', '\$dato', '\$fecha')"; \quad (2.2)$$

Para insertar datos en una tabla de la base de datos antes vista, se necesita ejecutar el comando anterior, primero comienza con el comando “insert into”, este comando es que da la instrucción de insertar los datos en la tabla, lo que prosigue es el nombre de la tabla en que vamos a insertar los datos, en este caso esta se llama “sensoresmonitoreo”, luego se indican los campos a llenar de la tabla (índice, valor, fecha), estos tienen que ser los mismos nombres que están en la tabla, después sigue el comando “values” con las variables que contienen los datos a ingresar, en este caso son (‘$sensor’, ‘$dato’, ‘$fecha’). Cabe destacar que los comandos (“insert into” y “values”) pueden ir en mayúsculas como en el ejemplo o en minúsculas, esto no varía sus resultados.

$$\$sql2 = "SELECT\ ID, TIPO, DESCRIPCION, RANGO, SETPOINT, ALARMA\ FROM\ \$user"; \quad (2.3)$$

La anterior línea corresponde a la obtención de datos de las tablas, esto puede ser de algunos datos específicos o de todos los datos al interior de esta. En este caso el comando inicial es “select” el cual puede venir acompañado del nombre de los campos a solicitar o acompañado de un asterisco (*), al poner este símbolo en vez de los datos, se obtendrán todos los campos de la tabla, luego se acompaña del comando “from” que permite seleccionar la tabla de la que se obtendrán los datos, en este ejemplo en vez de escribir el nombre de la tabla, se utilizó una variable que contiene el nombre de esta (“$user”).

$$\$sql3 = "UPDATE\ \$user\ SET\ SETPOINT = \$setpoint, ALARMA = '\$estado'\ WHERE\ ID = \$sensor"; \quad (2.4)$$

para poder modificar los datos de algunos o todos los campos de una tabla a elegir, se utiliza el comando “update”, este seguido del nombre de la tabla a editar y el comando “set”. Después del comando se agregan los campos a llenar y se igualan con la variable que tendrá ese campo, esto se hace para cada uno de los campos que se desean llenar y se separan por medio de una coma. Para acotar las ediciones o búsquedas se utiliza el comando “where”, este permite seleccionar solo los datos que cumplan las condiciones que le siguen, en este caso es que el campo “ID” tiene que ser igual a la variable “$sensor”, por ende, todas las filas de la tabla que cumplan esa condición serán modificadas.

$$mysqli\_query(\$con, \$sql) \quad (2.5)$$

“mysqli_query” es el comando que permite hacer las consultas, sea el caso de editar, seleccionar o inserta el contenido en una tabla de la base de datos, este requiere de solo dos variables, la primera es la conexión a la base de datos (2.1), y la segunda es la petición a la base de datos (2.2 o 2.3 o 2.4). este comando devuelve un “true” si se logró hacer la petición o un “false” si no se logró.

$$mysqli\_close(\$con); \quad (2.6)$$

Por último, el comando “mysqli_close” cierra la conexión a la base de datos, y necesita solo la variable de conexión a esta.

2.7. Programación aplicación Android.

La programación de la aplicación para smartphone se realizará en la plataforma “Appinventor” creada por estudiantes del MIT, esta plataforma permite crear aplicaciones para sistema operativo Android, esto de forma sencilla e intuitiva. Cabe destacar que esta plataforma se encuentra en internet y permite crear estas aplicaciones en línea y una vez finalizada descargar la app al ordenador. Como la aplicación no consta de funciones tan complejas, se utilizará esta plataforma, puesto que, esta plataforma permite realizar la mayoría de los requerimientos de la aplicación. Ha esta plataforma se puede ingresar a través de una cuenta de Gmail y es de uso gratuito.

Las funciones que tendrá la aplicación, son las siguientes: un login de ingreso, sección sensores activos, sección alarmas, sección gráficos, sección tablas y sección ajustes. A continuación, se detallarán las funciones de cada una de estas secciones.

2.7.1. Login.

En esta sección el usuario ingresará a la aplicación, a través de un usuario que será su correo electrónico y la claves que se le dará a él. Cabe mencionar que a través de la aplicación no se podrá registrar, esta es solo para ingresar, con esta medida se limita la cantidad de usuarios.

Fig. 2.19 Imagen tomada de plataforma App Inventor sección login aplicación.

Como se puede apreciar en la imagen anterior esta ventana consta de 3 componentes claves, que son los campos de texto usuario y clave, además de un botón ingresar que no se logra apreciar en la imagen puesto se encuentra más abajo y la estructura de la ventana no deja mostrarlo. Esta pantalla tiene como función tomar los datos de los campos al presionar el botón ingreso, al ocurrir esto se envían estos datos a un script en la página web que consultara a la base de datos de clientes si el usuario existe y si la clave coincide,

si esto ocurre se ingresara a la aplicación, de lo contrario se desplegara un mensaje de error.

2.7.2. Sensores.

Al ingresar correctamente la clave y el usuario, se llega a la pantalla de inicio que consta de un menú desplegable, en este menú la primera sección es la de los sensores, al seleccionar está en el menú se mostrara una tabla con todos los sensores que tenga el usuario.

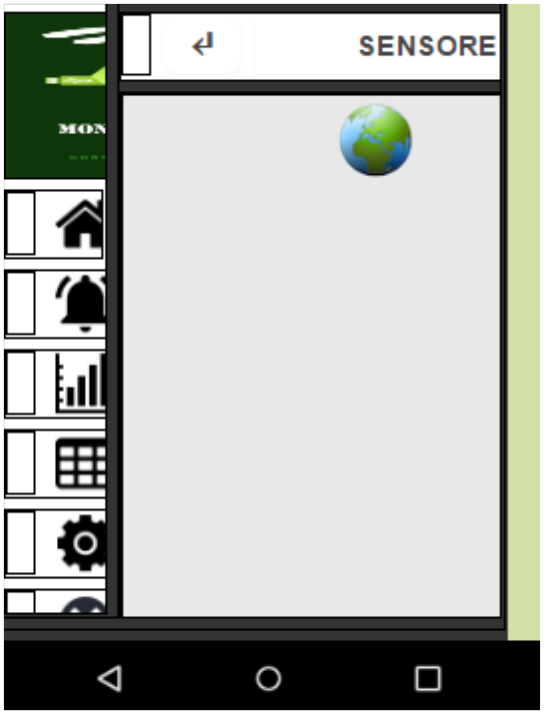


Fig. 2.20 Imagen tomada de plataforma App Inventor sección sensores aplicación.

Esta sección como se puede notar en la imagen anterior, consta de un objeto de navegación web, este permite mostrar la página que se inserte en esta, en ese espacio se muestra una tabla con los sensores y sus características. A esta página de navegación se envían los datos de usuario vía GET, para acceder a su base de datos correspondiente.

2.7.3. Alarmas.

Esta parte de la aplicación permite la activación de alarmas de todos los dispositivos, para la activación de estas, se poseen 4 elementos, estos son un selector de id del dispositivo, un cuadro de texto en el cual insertar el set-point con el cual se disparará la alarma, y los dos últimos elementos se reflejan 2 botones (on y off), al presionar alguno de ellos se apagará o encenderá la alarma. A continuación, se muestra la ventana básica de esta sección.

Fig. 2.21 Imagen tomada de plataforma App Inventor sección alarmas aplicación.

Debajo de los elementos para editar las alarmas, se encuentra un elemento de navegación en donde se verá el contenido de esa página web, este contenido es la misma tabla presente en sensores para identificar los cambios en la base de datos.

2.7.4. Gráficos.

Esta sección permite visualizar los datos de los sensores a través de gráficos entendibles, en esta pantalla se le pide al usuario la fecha y la id del sensor a graficar.

Fig. 2.22 Imagen tomada de plataforma App Inventor sección gráficos aplicación.

En esta pestaña de la aplicación se envían los datos fecha e id a un script en la página web para poder generar el grafico correspondiente.

2.7.5. Tablas.

La sección tablas pide al usuario los mismos datos utilizados en la sección gráficos, funciona de forma similar, pero en vez de graficar toma todos los datos del sensor requerido y tabularlos.

Fig. 2.23 Imagen tomada de plataforma App Inventor sección tablas aplicación.

2.7.6. Ajustes.

La última parte de la aplicación es la sección ajustes, esta permite hacer un cambio en la contraseña, esto para evitar que el usuario olvide la contraseña que se le otorgo, cambiando esta por una clave que el elija y recuerde de mejor manera. Posee 2 campos de ingreso de texto en los cuales se pide ingresar la contraseña actual y la contraseña nueva, si la contraseña actual concuerda con la entregada, se cambia por la contraseña nueva, de lo contrario se mostrará un mensaje de error. Además, esta sección consta de un botón para cerrar sesión y volver a la pestaña de login.

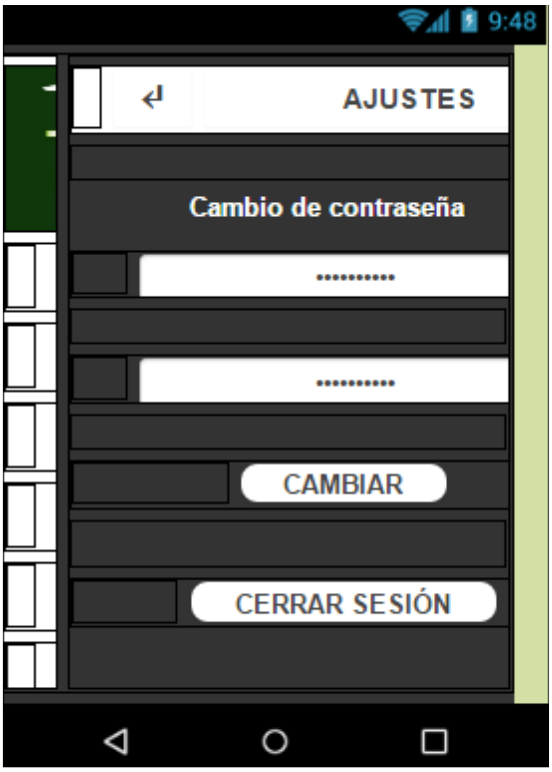


Fig. 2.24 Imagen tomada de plataforma App Inventor sección ajustes aplicación.

Capítulo 3. Resultados

En el capítulo resultados se identificarán los programas utilizados para la programación del sistema de monitoreo, tanto como la programación de los dispositivos, que es la programación basada en Arduino como la basada en Python, correspondientes a la Raspberry pi y Nodemcu respectivamente. También la programación de acceso a las bases de datos basadas en scripts en lenguaje PHP, y por último la programación de la aplicación Android que será a través de la plataforma App Inventor del MIT. Cabe destacar que para la programación del Nodemcu y Raspberry pi, se utilizaron script de ejemplos de páginas web y repositorios de librerías, los cuales se encuentran en la bibliografía.

3.1. Programas dispositivos lector de sensores.

Para la programación del dispositivo lector de sensores se utilizará el programa Arduino, que gracias a las librerías públicas se puede programar de forma idéntica a un Arduino normal, el dispositivo NodeMcu.
A continuación, se muestra el script Arduino segmentado en partes para poder analizar las partes más importantes de este.

Comenzando con las librerías y las variables a utilizar.

```
#include <Wire.h>
#include <Adafruit_ADS1015.h>
#include <ESP8266WiFi.h>
Adafruit_ADS1115 ads;
const char* ssid     = "Depto 1011";
const char* password = "departamento14dejulio";
int lapso=0;
const char* host = "monitoreolocal.com";
```

Se utilizan 3 librerías para este programa, la primera es utilizada para comunicar a la placa dispositivos que trabajan con el protocolo I2C/TWI, en este caso esta librería se utiliza

para comunicar el módulo conversor análogo digital a la placa. La segunda librería corresponde al módulo conversor análogo/digital, permitiendo utilizar todos los comandos para la obtención de las señales analógicas provenientes de los sensores. La última librería permite que toda la programación en el programa Arduino sea compatible con la placa Nodemcu. También se crear un objeto Adafruit, el cual permite tomar los valores que recibe el módulo.

El resto de líneas presentes corresponden a las variables a utilizar en el programa, las cuales son “ssid”, “password”, “host”, las cuales son: el nombre de la red wifi, la clave de la red wifi, el tiempo de sensado de los sensores y la pagina a la que serán enviados los datos respectivamente.

Dentro del “void setup()”, las líneas más importantes son las siguientes:

```
void setup()
{
 ads.setGain(GAIN_ONE);
 Serial.begin(115200);
 WiFi.begin(ssid, password);

 while (WiFi.status() != WL_CONNECTED) {
 delay(500);
 Serial.print(".");
 ads.begin();
 }
}
```

Como en todo programa de Arduino se comienza iniciando la comunicación serial, en este caso de 115200, también se inicia la comunicación wifi, con el comando WiFi.begin, el cual recibe las variables del nombre de la red wifi y su clave, luego de esto se crea un ciclo while que se ejecutara cada medio segundo mientras no se conecte a la red wifi, esto se reiterar hasta que se logre una conexión exitosa, por último se inicializa la comunicación con el módulo análogo/digital.

```
void loop() {
  int16_t adc0, adc1, adc2, adc3;
  adc0 = ads.readADC_SingleEnded(0);
  adc1 = ads.readADC_SingleEnded(1);
  adc2 = ads.readADC_SingleEnded(2);
  adc3 = ads.readADC_SingleEnded(3);
  lapso = analogRead(A0)*1000;
```

Luego del “setup” viene el “loop” que se ejecutara cíclicamente mientras este encendido el dispositivo, los primeros parámetros de este son, la inicialización de las variables como variables de 16 bits, estas son: adc0, adc1, adc2 y adc3, luego a estas se le otorgan los valores leídos por el módulo “ads1115”, también se lee el valor del potenciómetro que está conectado a la entrada analógica “A0”, siendo este valor multiplicado inmediatamente por 1000, teniendo esta variable un tiempo que va de los cero segundos a los 1024 segundos (17 minutos aprox.).

```
  String cadena1="";
  String cadena2="";
  String cadena3="";
  String cadena4="";
  cadena1=cadena1+adc0;
  cadena2=cadena2+adc1;
  cadena3=cadena3+adc2;
  cadena4=cadena4+adc3;
```

Luego el “loop” continua con la declaración de 4 variables tipo string inicializadas como vacías, esto para que al momento de sumarlas a las variables que contienen los valores leídos por el módulo análogo/digital, sean convertidos a variables del tipo string.

```
WiFiClient client;
const int httpPort = 80;
if (!client.connect(host, httpPort)) {
    Serial.println("connection failed");
    return;}
```

En esta parte del código, se crea la variable “client” del tipo WiFiClient y después una constante del tipo int httpPort, luego de esto se crea una condición, en la cual se pregunta si se pudo conectar al host, si se conecta se sigue con el programa, de lo contrario se muestra un mensaje de conexión fallida y se reinicia el ciclo “loop”.

```
String url = "/monitoreolocal.com/michaelcarlos/conexion2.php";
String key = "?clave=mcvv9024";
String dato1 = "&id=1";
String dato2 = "&sensor1=";
String dato3 = "&sensor2=";
String dato4 = "&sensor3=";
String dato5 = "&sensor4=";
```

Continuando con el programa se declaran 7 variables del tipo string, en las dos primeras se identifican la url de la página a enviar los datos y una clave para identificar al usuario, luego se declaran las 4 variables de a recibir por la página, estas son: el valor de la id del cliente y el valor de los 4 sensores.

```
client.print(String("GET ") + url + key + dato1 + dato2 + cadena1 + dato3 + cadena2 + dato4 + cadena3 + dato5 + cadena4 +" HTTP/1.1\r\n" + "Host: " + host + "\r\n" + "Connection: close\r\n\r\n");
unsigned long timeout = millis();
```

Con el código anterior se envía la petición del tipo “GET”, con los datos y la dirección de la página, luego se crea una variable tipo long, para utilizarla en el paso siguiente.

```
while (client.available() == 0) {
if (millis() - timeout > 5000) {
    Serial.println(">>> Client Timeout !");
    client.stop();
    return;
  }
}
```

En el paso anterior se crea un ciclo while que dura hasta que se realice una conexión exitosa, dentro de este ciclo se encuentra una condicionante que reiniciara el ciclo “loop” si la conexión se demora 5 más segundos.

```
  while (client.available()) {
    String line = client.readStringUntil('\r');
    Serial.print(line);
  }
delay(lapso);
}
```

Por último, se crea otro ciclo while en el que se leerá el contenido de respuesta de la página web consultada, y como último elemento del programa se crea un delay que en su interior se ingresa el valor de la variable lapso, detendrá el programa en esa cantidad de milisegundos.

3.2. Programa dispositivo de conteo.

Para la programación del dispositivo de conteo se utilizará la placa Raspberry pi Zero W, esta placa posee la cualidad de tener conexión a internet por medio de wifi. La programación se realizará por medio de un script Python. Las partes más importantes del programa se mostrarán y se explicarán a continuación.

```
import urllib
from datetime import datetime
import time
import Adafruit_ADS1x15
```

Primero se importan las librerías a utilizar, con estas se darán acceso a las funciones de edición y lectura de documentos de texto, a las funciones de tiempo y fecha y las funciones de lectura del módulo análogo/digital.

```
class enviar_datos:
```

```
        def conectar(self,host,campo,valor):
        self.variables=[]
        self.valores=[]
        self.campo = campo
        self.valor = valor
        self.host = host
        self.datos = {}
        for campo_variables,valor_variables in
zip(self.campo.split(";"),self.valor.split(";")):
                self.variables.append(campo_variables)
                self.valores.append(valor_variables)
        for variable,valor in zip(self.variables,self.valores):
                self.datos['%s'%variable] = valor
                try:
                return urllib.urlopen(self.host,urllib.urlencode(self.datos)).read()
        except:
                return "No se puede conectar a %s"%(self.host)
```

Al principio se crea la clase “enviar_datos”, en la cual se crea la función “conectar”, en la cual se ingresan las variables de host, campo y valor. Esta función permite almacenar datos en un formato idóneo para ser enviados a la página mediante vía “POST”, una vez almacenados se envían mediante la librería urllib.

```
url= 'http://www.monitoreolocal.com/monitoreolocal/contador/count.php'
variables= 'dato;hora;fecha;identificador;empresa'
#crear ADS1115 ADC (16-bit)
adc = Adafruit_ADS1x15.DAS1115()
GAIN = 1
```

Luego se llenan las 2 primeras variables con la dirección de la página web de destino y las variables que serán enviadas vía “POST”. También se crea una variable para la conexión del módulo análogo digital y se fija la variable “GAIN” en 1.

```
fichero = open('dato.txt')
var = fichero.read()
fichero.close()
j = int (var)
k = 0
emp = 1
```

Después se abre el archivo de texto “dato.txt”, en el cual se almacena el número de productos que ha contado el dispositivo, se lee ese valor y se le asigna a la variable “var”, posteriormente se cierra el archivo y se le asigna ese valor a la variable “j” y se convierte en tipo entero, también se definen 2 variables (k y emp), la cuales son respectivamente, la variable de identificación de paso y el numero identificador del cliente.

```
while True:
    values = [0]*4
    for i in range(4):
        values[i] = adc.read_adc(i, gain= GAIN)

    if (values[0] > 1000):
        archivo=open('dato.txt','w')
        archivo.write('0')
        archivo.close()
        j = 0
    if (values[1] > 10000 and k == 0:
        j+=1
        archivo=open('dato.txt','w')
        archivo.write(str(j))
        archivo.close()
        k = 5
        fecha=time.strftime("%Y-%m-%d")
        centesimas=int(time.time() * 100)
        cen = str(centesimas % 100).zfill(2)
        hora=time.strftime("%H:%M:%S.")
        valores=str(j)+";"+hora+str(cen)+";"+fecha+";"+str(verf)+";"+str(emp)
```

```
        conec = enviar_datos()
        print conec.conectar(url,variables,valores)
    else:
        k=0
```

La parte final del programa es un ciclo while, el cual se ejecutará siempre mientras el dispositivo este encendido. Este comienza con creando una variable del tipo arreglo con 4 espacios, luego se crea un ciclo for que se ejecutara 4 veces y lee los valores de los sensores conectados al módulo análogo/digital y se le asignan a cada uno de los espacios del arreglo “values”.

Luego se crea una condición en la cual se pide que el valor del sensor de proximidad, que está en el espacio número dos de la variable “values”, sea menor a 10000 y que la variable “k” sea igual a 0, si estas dos se cumplen se le suma un uno a la variable “j”, luego se guarda este nuevo número en el archivo “dato.txt”, se le da un valor 5 a la variable “k”, con esto se evita que se cuente más de una vez si el valor del sensor es menor a 10000. Luego se asignan las variables de fecha, hora y centésimas, luego se crea una variable “valores” en la cual se agregan todos los datos (contador, fecha, hora y centésimas), por último, se crea una variable “conec” con la clase “enviar_datos”, para luego utilizarla con la función conectar, insertando en esta las variables url, valores y variables. Por si no se cumplen las condiciones del “if” se ejecuta el “else” que da el valor 5 a la variable “k”.

Cabe destacar que ese programa va de la mano con un archivo de texto con el nombre de “dato.txt”, en el cual se guarda el número de conteo del dispositivo, esto para que cuando el dispositivo se pague no se reinicie el conteo. También en la primera entrada del módulo análogo/digital se encuentra conectado un pulsador, el cual al ser presionado cambia el valor del contador en el archivo “dato.txt”, por un 0, además de dar este valor a la variable “j”.

3.3. Programación scripts PHP página web.

Los scripts utilizados para el sistema de monitoreo se dividen en 3 grupos, el primero corresponde a los utilizados para él envió de datos desde los dispositivos a las bases de datos, el segundo grupo corresponde a los script de comunicación entre la aplicación para smartphone y las bases de datos, este grupo recibe y envía datos mediante vía "POST", el último grupo es el que permite visualizar el contenido del script y recibe los datos mediante vía "GET", esto últimos permiten ayudar a la aplicación a ejecutar funciones que no soporta de manera idónea la plataforma Appinventor. Cabe destacar que todos estos programas estarán en la sección Anexos al final del documento.

3.3.1. Programa recepción datos de dispositivos.

Este es un único programa llamado "count.php", el cual se vio en el capítulo anterior, este permite recibir los datos de los sensores y enviarlos a la base de datos "sensores monitoreo" la cual guarda estos datos por orden de llegada.

3.3.2. Programas comunicación base de datos con aplicación.

Estos programas como se dijo anteriormente, permiten recibir los datos y peticiones enviadas por medio de la aplicación Android y enviarles una respuesta en forma de variables simples.

El primero y único de estos scripts es el "loginapp.php", el cual recibe los datos de usuario y contraseña enviados desde la app, este verifica si existe este usuario y si la clave es correcta, si esta todo bien devuelve los datos de id del usuario, nombre, apellido, empresa y cantidad de sensores que posee, de lo contrario no envía todos estos datos.

3.3.3. Programas de visualización de contenidos de script.

En este grupo se encuentran la mayoría de todos los scripts utilizados para el sistema de monitoreo. En este caso se agruparán por uso en las funciones de la aplicación. Cabe destacar que el uso de estos es para dos acciones en concreto, la primera es para tabular de manera más ordenada los datos a mostrar y la segunda para realizar visualizar gráficos de manera más sencilla.

El script utilizado para la sección “sensores” en la aplicación es solamente “sensoresapp2.php”, el cual recibe solo dos datos de la aplicación, los cuales son la empresa del usuario y la cantidad de sensores que posee el usuario, con estos datos se consulta la base de datos del cliente y se visualiza en la página esta tabla.

El script para la sección “alarmas”, se utiliza el script “alarmasapp.php”, el cual recibe 5 datos mediante “GET” que son: la empresa, la id del sensor, el estado (on u off) y el set-point, con la id y el nombre de la empresa se identifica la línea del sensor a editar y se modifican los datos de estado y set-point, por los enviados a través de la app, luego se tabulan todos los sensores en la página para demostrar que se realizaron los cambios.

Se utilizan 3 script para la sección “gráficos”, estos son: “graficosapp.php”, “linealapp.php” y “linealapp2.php”, el primero de estos es el principal y es el que recibe los datos desde la aplicación, estos datos son: la empresa, el id del usuario, el id del sensor y la fecha de muestreo. En primer lugar con el nombre de la empresa y la id del sensor se consulta el tipo de este, si el tipo es “contador”, se ejecuta el script “linealapp2.php”, el cual permite graficar el lapsus de tiempo entre un producto y el otro con respecto a la hora exacta en que paso el producto, si por el contrario no es del tipo “contador”, se ejecuta el script “linealapp.php”, el cual permite graficar el valor del sensor contra la hora en que se tomó el dato. Los dos últimos scripts son funciones que permite graficar los datos en el script inicial. Para la realización de estos gráficos se utilizó la librería “Plotly.sj”, de la cual se tomaron sus ejemplos y se modificaron para que se adaptaran a las necesidades de la aplicación, esta librería se encuentra en la bibliografía con una página web de su uso.

El ultimo script utilizado es el “tablaapp.php”, el cual se encuentra en la sección “tablas” de la aplicación, este script recibe los campos de id del usuario, id del sensor y fecha de muestreo, con estos datos este script toma todos los datos del sensor seleccionado que se tomaron ese día y los tabula en una tabla, esta es visible a través de la aplicación.

3.4. Programación aplicación Android.

La programación de la app, como se explicó en el capítulo anterior se realizará a través de la plataforma App Inventor, esta plataforma consta de 2 interfaces para la programación de aplicaciones Android, la primera es la sección “diseñador”, en esta se

pueden arrastrar y ubicar los objetos a utilizar en la aplicación, cabe destacar que se crea la aplicación por mediante de la creación de ventanas las cuales pertenecerán a cada ventana que aparezca en la app.

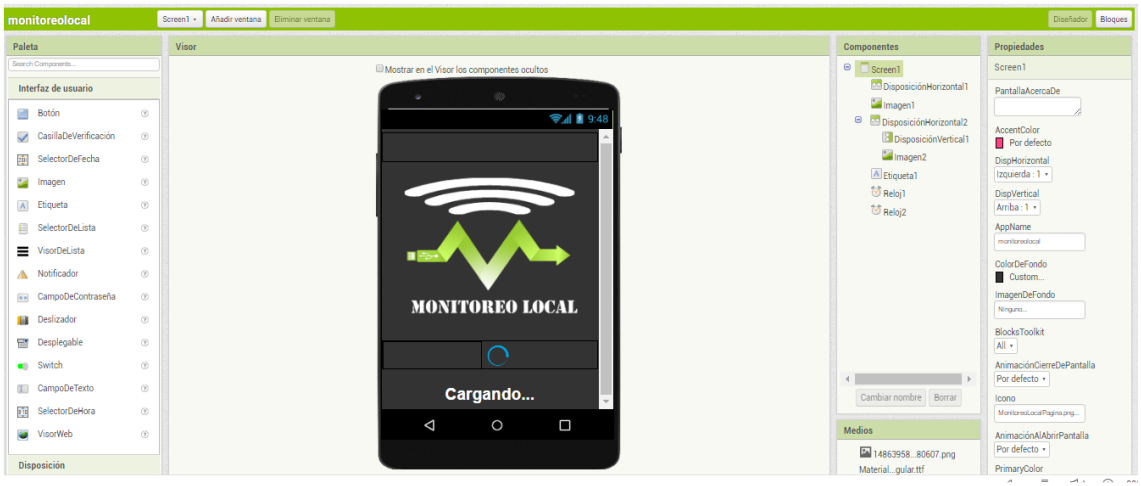


Fig. 3.1 Pantalla sección diseñador App Inventor.

En la imagen anterior se puede notar la interfaz de la sección diseñador, en esta sección se pueden colocar objetos como, por ejemplo: cuadros de texto, campos de contraseñas, botones, visor web, switch, imágenes, etc., también se pueden agregar otros correspondientes a sensores, mapas, multimedia, web, etc., en el caso de esta aplicación los más utilizados fueron los capos de texto, etiquetas, botones, visor web, web, imágenes, reloj y base de datos. En la parte derecha de la imagen se encuentran una lista de todos los objetos que han sido arrastrados a la aplicación y al lado de ella se encuentran todas las características editables del objeto en que se haga clic, en el caso de la imagen son las características de una “disposición vertical”. Al lado izquierdo se encuentran todos los componentes que se pueden utilizar en la aplicación, la forma de utilizarlos es arrastrarlos a la imagen del centro, la imagen del centro corresponde a una previsualización de la aplicación en ese momento.

La segunda sección corresponde a la opción bloques, es esta en donde se realiza la programación de los componentes de la app, la programación se realiza a través de bloques de programación los cuales se conectan entre sí para generar funciones para la aplicación.

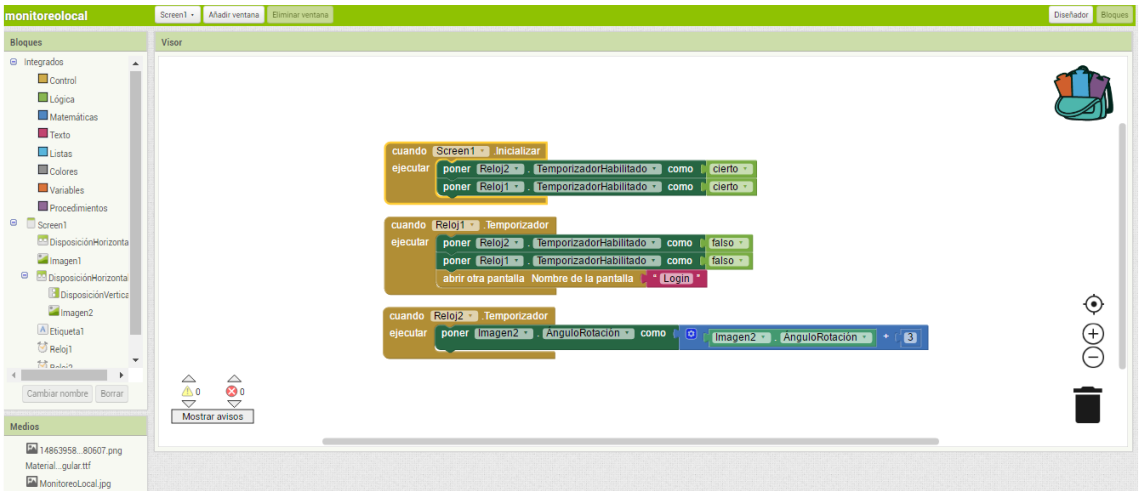


Fig. 3.2 Pantalla sección bloques Appinventor.

La imagen vista recientemente corresponde a la sección “bloques”, en esta se pueden encontrar todos los bloques necesarios para la programación, estos se encuentran en la parte izquierda de la imagen, estos constan de bloques estándares, los cuales son del tipo: control, lógica, matemáticas, textos, listas, colores, variables, procedimientos. El resto de los bloques que aparecen al inferior de estos son los bloques correspondientes a los objetos que se arrastraron a la aplicación en la sección diseñador, cada objeto que se incluye tiene su propio conjunto de bloques los cuales se pueden unir a los estándares o a otros objetos. En la imagen central se puede ver la programación por bloques de la ventana de inicio de la aplicación, esta consiste en tres partes, el primero bloque se ejecuta al iniciar la aplicación, y pone en marcha dos temporizadores, el tercer bloque corresponde a que cuando el segundo temporizador llegue a su intervalo se gire una imagen 3 grados, y el segundo bloque dice que si se llega al intervalo del primer temporizador se apaguen ambos y se abra la pantalla de login.

Eso es en general para entender un poco la forma en que se programó la aplicación, para la aplicación se crearon 10 ventanas, la primera es la pantalla de inicio de la imagen anterior.

Fig. 3.3 Imagen pantalla de inicio aplicación**.**

En la imagen anterior se puede ver el resultado de la primera ventana de la aplicación, esta pantalla dura activa 5 segundo y luego pasa a la pantalla de logeo.

Fig. 3.4 Imagen pantalla de login de la aplicación**.**

La pantalla de login permite el acceso a la app, ingresando el correo electrónico y una clave única, cabe destacar que la aplicación no permite registro de personas.

Fig. 3.5 Imagen pantalla de bienvenida y menú desplegable aplicación**.**

La aplicación consta de un menú desplegable, este permite tener acceso a las diferentes funciones de la aplicación en la mayoría de las ventanas de esta app. Los componentes del menú son: sensores, alarmas, gráficos, tablas, ajustes y salir, la opción salir corresponde a cerrar la aplicación.

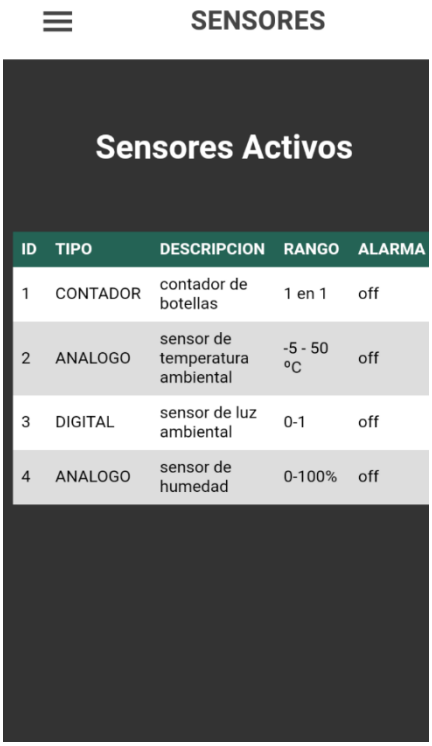


Fig. 3.6 Imagen pantalla sensores aplicación**.**

En esta imagen se permiten ver cada uno de los sensores que posee el cliente además de su tipo, descripción, rango en que trabaja y si esta activa su alarma o no. Esta pantalla muestra directamente a través del objeto visualizador web, lo que se encuentra en el script "sensoresapp.php", esto como ya se explicó para reducir la complejidad de la aplicación y dejar una estética un poco mejor que con las herramientas disponibles en la plataforma.

Fig. 3.7 Imagen pantalla de alarmas aplicación**.**

En esta ventana se puede ver en si como activar o desactivar las alarmas, primero se selecciona la id del sensor a configurar, luego se ingresa el set-point, luego de estos pasos, si se desea activar la alarma se presiona el botón verde “on”, de lo contrario si se desea apagar esta, se presiona el botón rojo “off”, al presionar alguno de estos dos botones se actualizarán automáticamente los campos set point y alarma del sensor seleccionado.

Fig. 3.8 Imagen pantalla selección de grafico aplicación**.**

En esta pantalla se ingresan el id del sensor y la fecha del día que se quiere graficar, luego de ingresar ambos datos se presiona el botón "enviar", el cual abre la ventana "graficos_elegidos", que es la imagen que se muestra a continuación.

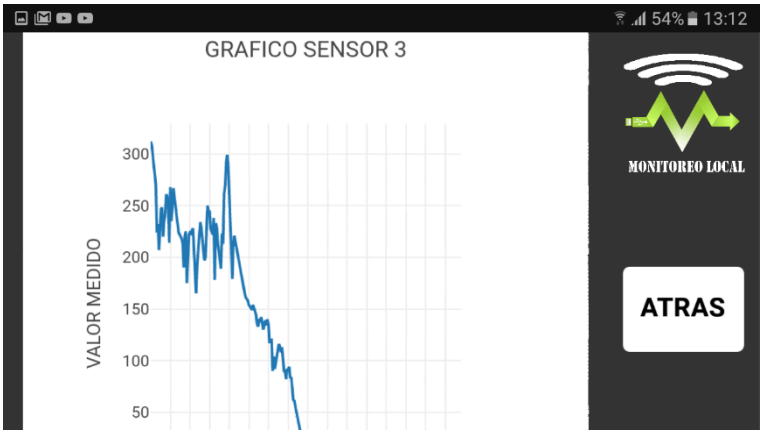


Fig. 3.9 Imagen pantalla graficos_elegidos aplicación**.**

En esta se puede apreciar el grafico compuesto por los datos obtenidos a través de la base de datos de los sensores, en esta pantalla se utilizó el elemento visor web, puesto que el grafico fue creado a través de un script PHP. Cabe mencionar que esta pantalla y la

pantalla “tablas_elegidas”, son las únicas ventanas que se encuentran en posición vertical y no poseen el menú desplegable, en reemplazo de este, se tiene un botón con el nombre de “atrás”, el cual permite regresar a la ventana anterior.

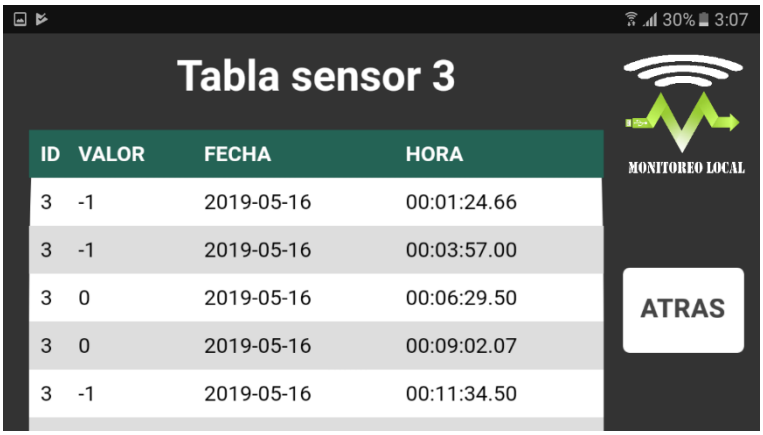


Fig. 3.10 Imagen pantalla tablas_elegidas aplicación**.**

Como en la ventana “graficos”, la ventana “tablas” es idéntica y pide los mismos datos, solo cambia entre ellas la ventana la cual abren, en este caso la ventana “tablas_elegidas”, en esta se muestran los datos del sensor y el día escogidos anteriormente, mostrando su id, el valor, la fecha y la hora en que se tomó el dato.

Fig. 3.11 Imagen pantalla ajustes aplicación**.**

Esta es la última pantalla de la aplicación y se pueden observar los campos de cambio de contraseña, que al llenarlos correctamente y apretar el botón cambiar, la contraseña será cambia en la base de datos por la que se eligió en ese momento. También se posee un botón llamado “cerrar sesión”, el cual permite borrar los datos del usuario de la app y así poder iniciar sesión con otra cuenta.

Capítulo 4. Conclusiones.

4.1. Introducción.

En este programa se ha presentado la implementación de un sistema de monitoreo de procesos en empresas. Se diseñaron dos tipos de dispositivos, el primero es un dispositivo de conteo de productos en líneas de producción, el segundo es un lector de sensores externos con salida analógica que va de 4 a 20 mA. También se construyó una aplicación Android en la cual se pueden ver los datos enviados por los dispositivos, en forma de tablas o gráficos, también con la capacidad de programar alarmas para cada uno de los dispositivos.

4.2. Conclusiones.

El diseño de dispositivos de monitoreo remoto es factible mientras se tenga una conexión a una red wifi en el lugar a monitorear, sin esta el sistema no funcionara, ya que, los dos dispositivos están basados en ese tipo de conexión.

El dispositivo contador funciona de manera óptima enviando datos con intervalos que no bajen de los 0.8 segundos, esto se debe a la programación del dispositivo, se probó también usar para este dispositivo un módulo Nodemcu, pero esta tenía lapsus de envío de datos superiores al segundo. Por eso se siguió utilizando la placa Raspberry Pi Zero, aunque su precio fuera mayor.

Un factor que afectaba la recepción de datos provenientes del dispositivo de conteo, fue la capacidad del hosting contratado, al principio para las pruebas se utilizó un hosting del tipo básico, pero la toma de datos hacia la base de datos se veía afectada y se detenía abruptamente, la solución fue cambiar este hosting por uno del tipo empresa, con este cambio ya no se volvieron a encontrar problemas al almacenar los datos.

Con respecto a la programación de la aplicación Android, se debe destacar que el uso de la plataforma App Inventor es una buena opción si se está empezando en la creación de aplicaciones, esta plataforma es muy intuitiva y permite a un usuario con nociones básicas de programación crear programas bastante útiles en poco tiempo y de manera sencilla.

Gracias al presente documento se puede apreciar que crear un sistema de monitoreo de procesos al interior de una empresa es factible y se comprobó con la creación de los dispositivos de monitoreo y la aplicación Android. Para mejorar este sistema, se tiene que saber que procesos se quieren monitorear para idear la forma idónea para sensarlo, en si no es complejo enviar los datos, en si es encontrar una forma con la cual el dispositivo interactúe con el proceso. En este caso la mayoría de los sensores ubicados en procesos claves poseen señales analógicas de 4 a 20 mA, y ya uno de estos dispositivos está hecho para estos casos. Para casos en que no se pueden medir variables analógicas, se diseñó como prueba el dispositivo de conteo, que ejemplifica un proceso no medible analógicamente.

Capítulo 5. Bibliografía

Vilches, J. M. (20 de 12 de 2019). *El 72% de los chilenos usa sus smartphones para trabajar y el 81% cree que aumenta su productividad*. Obtenido de Emol: https://www.emol.com/noticias/Tecnologia/2019/12/20/971111/Chilenos-sobre-celulares-inteligentes.html

Ditrendia. (2019). *Todas las estadísticas sobre móviles que deberías conocer 2019*. Obtenido de ditrendia digital marketing trends: https://mktefa.ditrendia.es/blog/todas-las-estad%C3%ADsticas-sobre-m%C3%B3viles-que-deber%C3%ADas-conocer-mwc19

Garita-Araya, R. A. (2013). Tecnología Móvil: desarrollo de sistemas y aplicaciones. *E-Ciencias de la Información*, 3.

Lucas, A. G. (s.f.). *EVOLUCION DE LAS APLICACIONES PARA MOVILES.*

sistemas.com. (10 de 11 de 2019). *sistemas.com*. Obtenido de sistemas.com: https://sistemas.com/aplicacion.php

MIT App Inventor 2. (2012). [Software online de creación de diseño de apps android]. Massachusetts Institute of Technology. https://appinventor.mit.edu

MCI Electronics - Electrónica para makers y profesionales DIY. (s. f.). MCI Electronics. Recuperado 22 de junio de 2020, de https://www.mcielectronics.cl/

Modulo Conversor Análogo Digital ADS1115. (s. f.). AFEL. Recuperado 22 de junio de 2020, de https://afel.cl/producto/modulo-conversor-analogo-digital-ads1115/

Módulo Conversor de Análogo Corriente a Voltaje 4-20mA a 0-5V. (s. f.). MaxElectrónica. Recuperado 22 de junio de 2020, de https://maxelectronica.cl/corriente/368-modulo-conversor-de-analogo-corriente-a-voltaje-4-20ma-a-0-5v.html

Sensor de proximidad infrarrojo grove | MCI Electronics.cl. (s. f.). MCI Electronics. Recuperado 22 de junio de 2020, de https://www.mcielectronics.cl/shop/product/sensor-de-proximidad-infrarrojo-grove-21771

Tarjeta Desarrollo NodeMCU V3 Amica Basada en el Módulo ESP8266 Wifi. (s. f.). MaxElectrónica. Recuperado 22 de junio de 2020, de https://maxelectronica.cl/microcontroladores/298-tarjeta-de-desarrollo-nodemcu-v3-amica-basada-en-el-modulo-esp8266-wifi.html

Transformador 5V micro USB compatible con Raspberry Pi 3 | MCI Electronics.cl. (s. f.). MCI Electronics. Recuperado 22 de junio de 2020, de https://www.mcielectronics.cl/shop/product/transformador-5v-micro-usb-compatible-con-raspberry-pi-3-24977?search=fuente+de+poder

Buy a Raspberry Pi Zero W – Raspberry Pi. (s. f.). Raspberry Pi Fundation. Recuperado 22 de junio de 2020, de https://www.raspberrypi.org/products/raspberry-pi-zero-w/

Arduino (1.8.3). (2018). [Software de progración de placas Arduino]. Arduino. https://www.arduino.cc

Python (2.7.0). (2010). [Software de programacion]. Python Software Foundation. https://www.python.org

adafruit/Adafruit_Python_ADS1x15. (2018, 7 diciembre). GitHub. https://github.com/adafruit/Adafruit_Python_ADS1x15

PHP: ¿Qué es PHP? - Manual. (s. f.). The PHP Group. Recuperado 22 de junio de 2020, de https://www.php.net/manual/es/intro-whatis.php

adafruit/Adafruit_ADS1X15. (s. f.). GitHub. Recuperado 22 de junio de 2020, de https://github.com/adafruit/Adafruit_ADS1X15

Enviar peticiones POST a sitios Web con Python. (2012, 24 enero). Cristalab. http://www.cristalab.com/tutoriales/enviar-peticiones-post-a-sitios-web-con-python-c103492l/

ESP8266: Subir datos a un servidor mediante WiFi. (2019, 4 octubre). Tienda y Tutoriales Arduino. https://www.prometec.net/esp8266-subir-valores/

plotly/plotly.js. (s. f.). GitHub. Recuperado 23 de junio de 2020, de https://github.com/plotly/plotly.js/

Shokeen, M. (2017, 18 septiembre). *Crea Gráficas Interactivas Usando Plotly.js, Parte 1: Comenzando*. Envato. https://code.tutsplus.com/es/tutorials/create-interactive-charts-using-plotlyjs-getting-started--cms-29029

Anexo A. Programa dispositivo sensores

```
#include <Wire.h>
#include <Adafruit_ADS1015.h>
#include <ESP8266WiFi.h>
Adafruit_ADS1115 ads;
const float multiplier = 0.1875F;
const char* ssid     = "Depto 1011";
const char* password = "xxxxxxxxxxxx";
int lapso=0;
const char* host = "monitoreolocal.com";
void setup()
{
 ads.setGain(GAIN_ONE);
 Serial.begin(115200);
 // We start by connecting to a WiFi network
 Serial.println("");
 Serial.println("");
 Serial.print("Connecting to ");
 Serial.println(ssid);

 WiFi.begin(ssid, password);

 while (WiFi.status() != WL_CONNECTED) {
 delay(500);
 Serial.print(".");
 ads.begin();
 }

 Serial.println("");
 Serial.println("WiFi connected");
 Serial.println("IP address: ");
 Serial.println(WiFi.localIP());
}

void loop() {
  int16_t adc0, adc1, adc2, adc3;


  adc0 = ads.readADC_SingleEnded(0);
  adc1 = ads.readADC_SingleEnded(1);
  adc2 = ads.readADC_SingleEnded(2);
  adc3 = ads.readADC_SingleEnded(3);
  lapso = analogRead(A0)*1000;

  String cadena1="";
  String cadena2="";
  String cadena3="";
  String cadena4="";
```

```
  cadena1=cadena1+adc0;
  cadena2=cadena2+adc1;
  cadena3=cadena3+adc2;
  cadena4=cadena4+adc3;
   Serial.print("connecting to ");
  Serial.println(host);

  // Use WiFiClient class to create TCP connections
  WiFiClient client;
  const int httpPort = 80;
  if (!client.connect(host, httpPort)) {
    Serial.println("connection failed");
    return;
  }

  // We now create a URL for the request
  String url = "/monitoreolocal.com/michaelcarlos/conexion2.php";
  String key = "?clave=mcvv9024";
  String dato1 = "&id=1";
  String dato2 = "&sensor1=";
  String dato3 = "&sensor2=";
  String dato4 = "&sensor3=";
  String dato5 = "&sensor4=";

  Serial.print("Requesting URL: ");
  Serial.println(url);

  // This will send the request to the server
  client.print(String("GET ") + url + key + dato1 + dato2 + cadena1 + dato3 + cadena2 +
dato4 + cadena3 + dato5 + cadena4 +" HTTP/1.1\r\n" +
               "Host: " + host + "\r\n" +
               "Connection: close\r\n\r\n");
  unsigned long timeout = millis();
  while (client.available() == 0) {
    if (millis() - timeout > 5000) {
      Serial.println(">>> Client Timeout !");
      client.stop();
      return;
    }
  }

  while (client.available()) {
    String line = client.readStringUntil('\r');
    Serial.print(line);
  }
  Serial.println();
  Serial.println("closing connection");

  delay(lapso);
}
```

Anexo B. Programa dispositivo contador

```
#!/usr/bin/python
# -*- coding: utf-8 -*-
import urllib
from datetime import datetime
import time
import Adafruit_ADS1x15

class enviar_datos:
        def conectar(self,host,campo,valor):
        self.variables=[]
        self.valores=[]
        self.campo = campo
        self.valor = valor
        self.host = host
        self.datos = {}
        for campo_variables,valor_variables in
zip(self.campo.split(";"),self.valor.split(";")):
                self.variables.append(campo_variables)
                self.valores.append(valor_variables)
        for variable,valor in zip(self.variables,self.valores):
                self.datos['%s'%variable] = valor
                try:
                return urllib.urlopen(self.host,urllib.urlencode(self.datos)).read()
        except:
                return "No se puede conectar a %s"%(self.host)

url= 'http://www.monitoreolocal.com/monitoreolocal/contador/count.php'
variables= 'dato;hora;fecha;identificador;empresa'
#crear ADS1115 ADC (16-bit)
adc = Adafruit_ADS1x15.DAS1115()

GAIN = 1

fichero = open('dato.txt')
var = fichero.read()
fichero.close()
j = int (var)
k = 0
verf = 1
emp = 1

while True:
   #se leen los 4 canales del modulo ADC
      values = [0]*4
      for i in range(4):
            #se lee cada uno de los valores
            values[i] = adc.read_adc(i, gain= GAIN)
```

```
if (values[0] > 1000):
        archivo=open('dato.txt','w')
        archivo.write('0')
        archivo.close()
        j = 0
if (values[1] > 10000 and k == 0:
        j+=1
        archivo=open('dato.txt','w')
        archivo.write(str(j))
        archivo.close()
        k = 5
        fecha=time.strftime("%Y-%m-%d")
        centesimas=int(time.time() * 100)
        cen = str(centesimas % 100).zfill(2)
        hora=time.strftime("%H:%M:%S.")
        valores=str(j)+";"+hora+str(cen)+";"+fecha+";"+str(verf)+";"+str(emp)
        conec = enviar_datos()
        print conec.conectar(url,variables,valores)
else:
        k=0
```

Anexo C. Script count.php

```
<?php
 $conn = new mysqli("localhost", "monitorx_michael", "xxxxxxxxxxx", "monitorx_monitoreolocal");
$sensor = $_POST["identificador"];
$dato = htmlspecialchars($_POST["dato"]);
$fecha = htmlspecialchars($_POST["fecha"]);
$hora = htmlspecialchars($_POST["hora"]);
$empresa = htmlspecialchars($_POST["empresa"]);

$sql = "INSERT INTO sensoresmonitoreo (empresa,indice,valor,fecha,hora) VALUES ('$empresa', '$sensor', '$dato', '$fecha', '$hora')";
if (mysqli_query($conn, $sql)) {
    echo "New record created successfully";
} else {
    echo "Error: " . $sql . "<br>" . mysqli_error($conn);
}
mysqli_close($conn);
echo $dato
?>
```

Anexo D. Script loginapp.php

```
<?php
$user = htmlspecialchars(($_POST['usuario']));
$pass = htmlspecialchars($_POST['clave']);
if(empty($user) || empty($pass)){
echo "error_login";
}
else{
$con = mysqli_connect("localhost", "genesist_usermaster", " xxxxxxxxxxx ", "genesist_appdual") or die("Sin Conexion");

$sql= "SELECT id, nombre, apellido, empresa, contra, sensores FROM clientes WHERE mail='$user'";

if (!$resultado = $con->query($sql)) {
    echo "Lo sentimos, este sitio web esta experimentando problemas.";
    echo "Error: La ejecucion de la consulta fallo debido a: \n";
    echo "Query: " . $sql . "\n";
    echo "Errno: " . $mysqli->errno . "\n";
    echo "Error: " . $mysqli->error . "\n";
    exit;
}

if ($resultado->num_rows === 0) {
    echo "Lo sentimos. No se pudo encontrar una coincidencia para el ID $aid. Intentelo de nuevo.";
    exit;
}

$clientes = $resultado->fetch_assoc();

if($clientes['contra'] == $pass){
    echo "login_ok";
    echo ";";
    echo $clientes['id'];
    echo ";";
    echo $clientes['empresa'];
    echo ";";
    echo $clientes['nombre'];
    echo ";";
    echo $clientes['apellido'];
    echo ";";
    echo $user;
    echo ";";
    echo $clientes['sensores'];
}
else{
  echo $pass;
```

```
  echo "     ";
  echo $id;
  echo "     ";
  echo $correo;
}
}
?>
```

Anexo E. Script sensoresapp2.php

```
<?php
$user = rtrim (htmlspecialchars(($_GET['usuario'])));
$sen = rtrim (htmlspecialchars($_GET['sensores']));
$O=1;

?>
<html>
   <style>
#content{
  overflow-x: hidden;
  white-space: nowrap;
}
</style>
<script>
function adjustZoom(){
   var newZoomCss =
'zoom:'+document.documentElement.clientWidth*100/content.scrollWidth+'%; ';
   var truncUpTo = 0;
   var firstNL = document.body.style.cssText.indexOf(';');
   if(firstNL && document.body.style.cssText.substr(0,5)==='zoom:'){
      truncUpTo = firstNL;
   }
   document.body.style.cssText = newZoomCss +
document.body.style.cssText.substr(truncUpTo);
}
window.addEventListener('resize', function(){
   adjustZoom();adjustZoom();adjustZoom();
});
window.addEventListener('load', function(){
   adjustZoom();adjustZoom();adjustZoom();
});
</script>
 <body>
    <div id="content">
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
<br></br>
         <center><h1 Style="color:#fff">Sensores Activos</h1></center><br></br>
         <link href="/carpeta/estilos/estilos2.css" rel="stylesheet" type="text/css">

         <table id="main-container3">

<thead>
<tr>
         <th>ID</th>
         <th>TIPO</th>
         <th>DESCRIPCION</th>
```

```
        <th>RANGO</th>
        <th>ALARMA</th>
</tr>
</thead>
<tbody>
<?php
while($O <= $sen+1){
   $con = mysqli_connect("localhost", "genesist_usermaster", " xxxxxxxxxxx", "genesist_appdual") or die("Sin Conexion");

$sql= "SELECT TIPO, DESCRIPCION, RANGO, ALARMA FROM $user WHERE ID='$O'";

if (!$resultado = $con->query($sql)) {
   echo "Lo sentimos, este sitio web esta experimentando problemas.";
   exit;
}

if ($resultado->num_rows === 0) {
   exit;
}

$senso = $resultado->fetch_assoc();
     echo "
                <tr >
                        <td>".($O)."</td>
                        <td>".$senso['TIPO']."</td>
                        <td>".$senso['DESCRIPCION']."</td>
                        <td>".$senso['RANGO']."</td>
                        <td>".$senso['ALARMA']."</td>
                </tr>";
   $O++;
   }
?>
</tbody>
</table>
</div>
</body>
</html>
?>
```

Anexo F. Script alarmasapp.php

```
<?php
$user = rtrim (htmlspecialchars(($_GET['usuario'])));
$fecha = rtrim (htmlspecialchars($_GET['fecha']));
$sensor = rtrim (htmlspecialchars($_GET['sensor']));
$estado = rtrim (htmlspecialchars(($_GET['estado'])));
$setpoint = rtrim (htmlspecialchars($_GET['setpoint']));
$user2=$user.'2';
$O=1;
if(empty($estado) || empty($setpoint)){
$O=1;
}
else{
   $con = mysqli_connect("localhost", "genesist_usermaster", " xxxxxxxxxxx", "genesist_appdual") or die("Sin Conexion");
$sql="UPDATE $user SET SETPOINT = $setpoint, ALARMA = '$estado' WHERE ID=$sensor";

if (!$resultado = $con->query($sql)) {
   echo "Lo sentimos, este sitio web esta experimentando problemas.";
   echo "Error: La ejecucion de la consulta fallo debido a: \n";
   echo "Query: " . $sql . "\n";
   echo "Errno: " . $mysqli->errno . "\n";
   echo "Error: " . $mysqli->error . "\n";
   exit;
}

if ($resultado->num_rows === 0) {
   echo "Lo sentimos. No se pudo encontrar una coincidencia para el ID $aid. Intentelo de nuevo.";
   exit;
}
$con2 = mysqli_connect("localhost", "genesist_usermaster", " xxxxxxxxxxx", "genesist_appdual") or die("Sin Conexion");
$sql2="UPDATE $user2 SET setpoint = $setpoint, alarma = '$estado' WHERE sensor=$sensor";

if (!$resultado2 = $con2->query($sql2)) {
   echo "Lo sentimos, este sitio web esta experimentando problemas.";
   echo "Error: La ejecucion de la consulta fallo debido a: \n";
   echo "Query: " . $sql2 . "\n";
   echo "Errno: " . $mysqli->errno . "\n";
   echo "Error: " . $mysqli->error . "\n";
   exit;
}

if ($resultado2->num_rows === 0) {
```

```
   echo "Lo sentimos. No se pudo encontrar una coincidencia para el ID $aid. Intentelo de nuevo.";
   exit;
}
}
?>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
<style>
#content{
  overflow-x: hidden;
  white-space: nowrap;
}
</style>
<script>
function adjustZoom(){
   var newZoomCss = 'zoom:'+document.documentElement.clientWidth*100/content.scrollWidth+'%; ';
   var truncUpTo = 0;
   var firstNL = document.body.style.cssText.indexOf(';');
   if(firstNL && document.body.style.cssText.substr(0,5)==='zoom:'){
      truncUpTo = firstNL;
   }
   document.body.style.cssText = newZoomCss + document.body.style.cssText.substr(truncUpTo);
}
window.addEventListener('resize', function(){
   adjustZoom();adjustZoom();adjustZoom();
});
window.addEventListener('load', function(){
   adjustZoom();adjustZoom();adjustZoom();
});
</script>
<div id="content">
<link href="/carpeta/estilos/estilos2.css" rel="stylesheet" type="text/css">
         <table id="main-container3">
         <link href="/carpeta/estilos/estilos2.css" rel="stylesheet" type="text/css">
         <table id="main-container3">
<thead>
<?php
$con = mysqli_connect("localhost", "genesist_usermaster", " xxxxxxxxxxx", "genesist_appdual") or die("Sin Conexion");
   $sql= "SELECT ID,TIPO,DESCRIPCION,RANGO,SETPOINT,ALARMA FROM $user";
   $result=mysqli_query($con,$sql);
?>
<tr>
         <th>ID</th>
         <th>TIPO</th>
         <th>DESCRIPCION</th>
         <th>RANGO</th>
```

```
        <th>SETPOINT</th>
        <th>ALARMA</th>
</tr>
</thead>
<tbody>
<?php
while ($ver=mysqli_fetch_row($result)){
    echo "
                <tr >
                        <td>".$ver[0]."</td>
                        <td>".$ver[1]."</td>
                        <td>".$ver[2]."</td>
                        <td>".$ver[3]."</td>
                        <td>".$ver[4]."</td>
                        <td>".$ver[5]."</td>
                </tr>";
  }
?>
</tbody>
</table>
</div>
```

Anexo G. Script graficosapp.php

```
<?php
$user = rtrim (htmlspecialchars(($_GET['usuario'])));
$id = rtrim (htmlspecialchars(($_GET['id'])));
$fecha = rtrim (htmlspecialchars($_GET['fecha']));
$sensor = rtrim (htmlspecialchars($_GET['sensor']));
$O=1;
$con = mysqli_connect("localhost", "genesist_usermaster", " xxxxxxxxxxx", "genesist_appdual") or die("Sin Conexion");
$sql= "SELECT TIPO FROM $user WHERE ID = '$sensor'";
if (!$resultado = $con->query($sql)) {
   echo "Lo sentimos, este sitio web esta experimentando problemas.";
   exit;
}

if ($resultado->num_rows === 0) {
   exit;
}
$senso = $resultado->fetch_assoc();
?>
      <link rel="stylesheet" type="text/css" href="js/bootstrap/css/bootstrap.css">
      <script src="js/jquery-1.7.2.min.js"></script>
      <script src="js/plotly-latest.min.js"></script>
<link href="/carpeta/estilos/estilos2.css" rel="stylesheet" type="text/css">
   <body style="background-color:#333333;">
      <center><h1 Style="color:#fff">Seleccionar Grafico</h1></center><br></br>
      <div class="container">
         <div class="row">
            <div class="col-sm-12">
               <div class="panel-primary">
                  <div class="panel panel-heading">
                     <center>graficos sensor <?php echo $sensor?></center>
                  </div>
                  <div class="panel panel-body">
                     <div class="row">
                        <div class="col-sm-12">
                           <div id="cargaLineal"></div>
                        </div>
                     </div>
                  </div>
               </div>
            </div>
         </div>
      </div>
   </body>

<br></br><br></br><br></br><br></br><br></br><br></br><br></br><br></br><br
></br><br></br>
```

```
<?php
echo $senso['TIPO'];
if($senso['TIPO']=='CONTADOR'){
$link='linealapp.php?usuario='.$user.'&id='.$id.'&fecha='.$fecha.'&sensor='.$sensor;
?>
<script type="text/javascript">
   $(document).ready(function(){
      $('#cargaLineal').load('linealapp.php?usuario=<?php echo $user?>&id=<?php echo
$id?>&fecha=<?php echo $fecha?>&sensor=<?php echo $sensor?>');
   });
</script>
<?php
}
else{

$link='linealapp2.php?usuario='.$user.'&id='.$id.'&fecha='.$fecha.'&sensor='.$sensor;
?>
<script type="text/javascript">
   $(document).ready(function(){
      $('#cargaLineal').load('linealapp2.php?usuario=<?php echo $user?>&id=<?php
echo $id?>&fecha=<?php echo $fecha?>&sensor=<?php echo $sensor?>');
   });
</script>
<?php
}
```

Anexo H. Script linealapp.php

```
<?php
$usuario = rtrim (htmlspecialchars(($_GET['usuario'])));
$id = rtrim (htmlspecialchars(($_GET['id'])));
$fecha = rtrim (htmlspecialchars($_GET['fecha']));
$sensor = rtrim (htmlspecialchars($_GET['sensor']));

function RestarHoras($horaini,$horafin)
{
	$horai=substr($horaini,0,2);
	$mini=substr($horaini,3,2);
	$segi=substr($horaini,6,2);
	$csegi=substr($horaini,9,2);

	$horaf=substr($horafin,0,2);
	$minf=substr($horafin,3,2);
	$segf=substr($horafin,6,2);
	$csegf=substr($horafin,9,2);

	$ini=((($horai*60)*60*100)+($mini*6000)+($segi*100)+$csegi);
	$fin=((($horaf*60)*60*100)+($minf*6000)+($segf*100)+$csegf);

	$dif=$fin-$ini;
	$difh=floor($dif/360000);
	$difm=floor(($dif-($difh*360000))/6000);
	$difs=floor(($dif-($difm*6000)-($difh*360000))/100);
	$difcs=$dif-($difs*100)-($difm*6000)-($difh*360000);
	if($difh>0){
	 return $difh.":".$difm.":".$difs.".".$difcs;
	}
	elseif($difm>0){
	   return $difm.":".$difs.".".$difcs;
	}
	elseif($difs>0){
	   if($difcs<10){
	   return $difs."."."0".$difcs;
	   }
	   else{
	      return $difs.".".$difcs;
	   }
	}
	else{
	    if($difcs<10){
	   return "0"."."."0".$difcs;
	   }
	   else{
	  return "0".".".$dif;
	   }
```

```
        }
}
   $con = mysqli_connect("localhost", "genesist_usermaster", "xxxxxxxxxxx", "genesist_appdual") or die("Sin Conexion");
   $sql= "SELECT valor, fecha, hora FROM sensoresmonitoreo WHERE (fecha = '$fecha' AND empresa = '$id' AND indice = '$sensor')";

   if (!$resultado = $con->query($sql)) {
   echo "Lo sentimos, este sitio web esta experimentando problemas.";
   exit;
}

if ($resultado->num_rows === 0) {
   echo 'malo';
}


   $valoresY=array();
   $valoresX=array();
   while ($ver=mysqli_fetch_row($resultado)){
      if($ver[0]=='1'){
         $valoresY[]='0';
         $valoresX[]=$ver[0];
         $varw=$ver[2];
      }
      else{
         $valoresX[]=$ver[0];
         $valoresY[]=RestarHoras($varw,$ver[2]);
         $varw=$ver[2];
      }
   }

   $datosX=json_encode($valoresX);
   $datosY=json_encode($valoresY);

?>
<div id="graficaLineal"></div>

<script type="text/javascript">
   function crearCadenaLineal(json){
      var parsed = JSON.parse(json);
      var arr = [];
      for(var x in parsed){
         arr.push(parsed[x]);
      }
      return arr;
   }
</script>

<script type="text/javascript">
```

```
  datosX=crearCadenaLineal('<?php echo $datosX ?>');
  datosY=crearCadenaLineal('<?php echo $datosY ?>');

  var trace1 = {
     x: datosX,
     y: datosY,
     type: 'scatter'
  };

  var data = [trace1];

  var layout = {
 title: 'GRAFICO DE VELOCIDAD DE PRODUCTOS (S) <?php echo $fecha?>',
 xaxis: {
  title: 'NUMERO DE PRODUCTO',
  showgrid: false,
  zeroline: false
 },
 yaxis: {
  title: 'VELOCIDAD DE PRODUCTO',
  showline: false
 }
};

  Plotly.newPlot('graficaLineal', data, layout);
</script>
```

Anexo I. Script linealapp2.php

```
<?php
$usuario = rtrim (htmlspecialchars(($_GET['usuario'])));
$id = rtrim (htmlspecialchars(($_GET['id'])));
$fecha = rtrim (htmlspecialchars($_GET['fecha']));
$sensor = rtrim (htmlspecialchars($_GET['sensor']));

   $con = mysqli_connect("localhost", "genesist_usermaster", "xxxxxxxxxxx", "genesist_appdual") or die("Sin Conexion");
   $sql= "SELECT valor, fecha, hora FROM sensoresmonitoreo WHERE (fecha = '$fecha' AND empresa = $id AND indice = $sensor)";

   if (!$resultado = $con->query($sql)) {
   echo "Lo sentimos, este sitio web esta experimentando problemas.";
   exit;
}

if ($resultado->num_rows === 0) {
   echo 'malo';
}


   $valoresY=array();
   $valoresX=array();
   while ($ver=mysqli_fetch_row($resultado)){

         $valoresX[]=$ver[2];
         $valoresY[]=$ver[0];
         $varw=$ver[2];
      }

   $datosX=json_encode($valoresX);
   $datosY=json_encode($valoresY);

?>
<div id="graficaLineal"></div>

<script type="text/javascript">
   function crearCadenaLineal(json){
      var parsed = JSON.parse(json);
      var arr = [];
      for(var x in parsed){
         arr.push(parsed[x]);
      }
      return arr;
   }
</script>
```

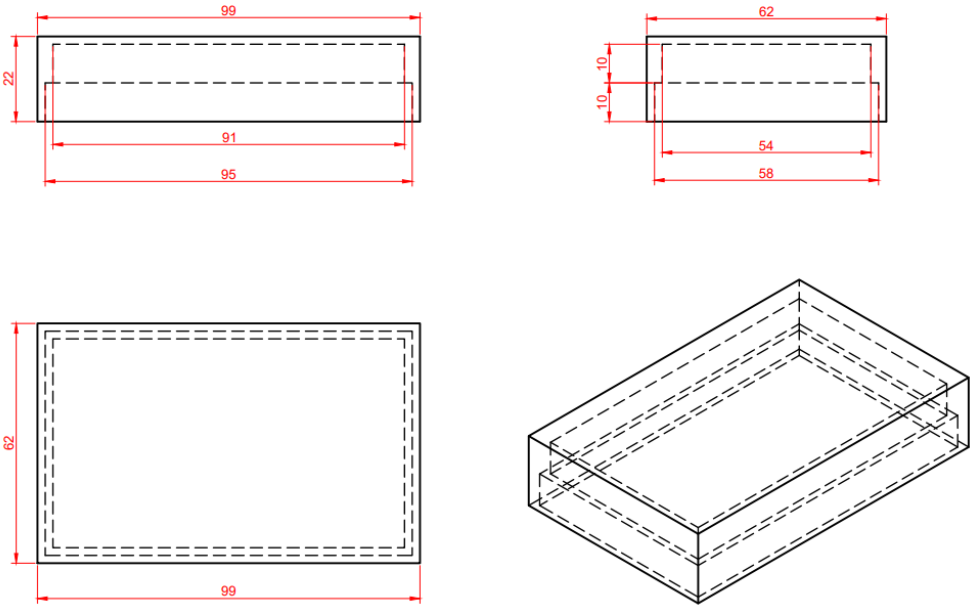
```
<script type="text/javascript">

    datosX=crearCadenaLineal('<?php echo $datosX ?>');
    datosY=crearCadenaLineal('<?php echo $datosY ?>');

    var trace1 = {
        x: datosX,
        y: datosY,
        type: 'scatter'
    };

    var data = [trace1];

    var layout = {
  title: 'GRAFICO SENSOR <?php echo $sensor?>',
  xaxis: {
    title: 'HORA DE SENSADO'
  },
  yaxis: {
    title: 'VALOR MEDIDO'
  }
};

    Plotly.newPlot('graficaLineal', data, layout);
</script>
```

Anexo J. Script tablaapp.php

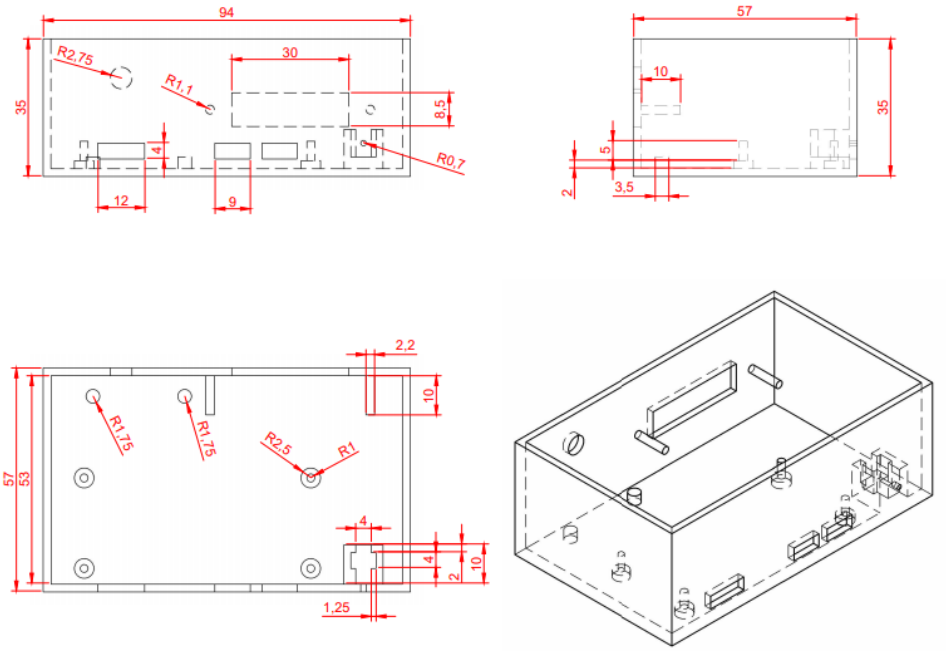
```
<?php
$user = rtrim (htmlspecialchars(($_GET['usuario'])));
$fecha = rtrim (htmlspecialchars($_GET['fecha']));
//echo $fecha;
$sensor = rtrim (htmlspecialchars($_GET['sensor']));
$O=1;
?>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />

<link href="/carpeta/estilos/estilos2.css" rel="stylesheet" type="text/css">
        <table id="main-container3">
        <link href="/carpeta/estilos/estilos2.css" rel="stylesheet" type="text/css">
        <table id="main-container3">

<thead>
<div class="panel panel-heading">
                    <center><h1 Style="color:#fff">Tabla sensor <?php echo $sensor?></center>
                </div>
<?php
$con = mysqli_connect("localhost", "genesist_usermaster", "xxxxxxxxxxx", "genesist_appdual") or die("Sin Conexion");
   $sql= "select valor, fecha, hora from sensoresmonitoreo where fecha = '$fecha' and empresa = '$user' and indice = '$sensor'";
   $result=mysqli_query($con,$sql);
?>
<tr>
        <th>ID</th>
        <th>VALOR</th>
        <th>FECHA</th>
        <th>HORA</th>
</tr>
</thead>
<tbody>
<?php
while ($ver=mysqli_fetch_row($result)){
    echo "
                <tr >
                        <td>".$sensor."</td>
                        <td>".$ver[0]."</td>
                        <td>".$ver[1]."</td>
                        <td>".$ver[2]."</td>
                </tr>";
   }
?>
</tbody>
</table>
```

Anexo K. Planos dispositivos de monitoreo.

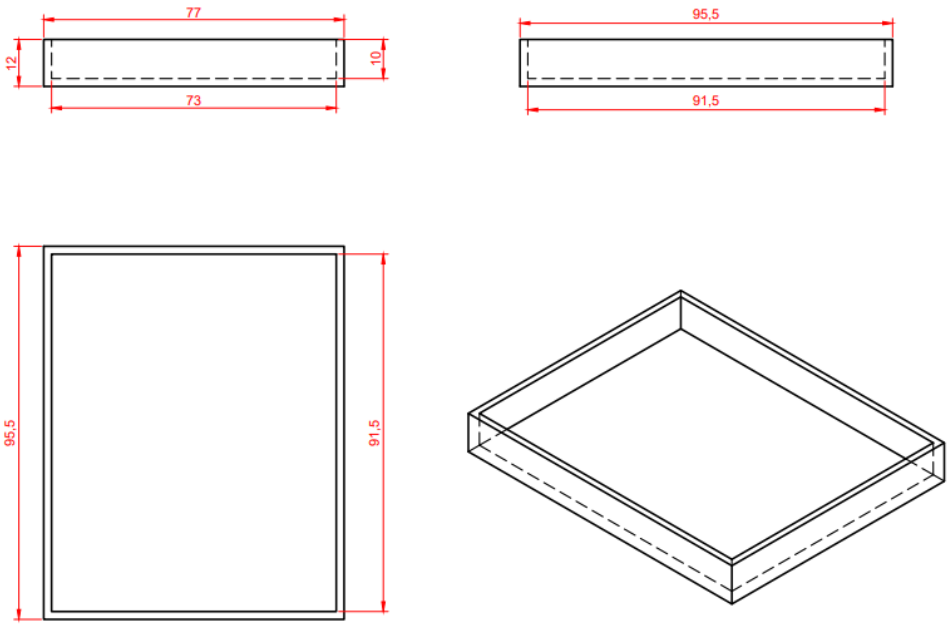
Tapa dispositivo contador



Carcasa dispositivo contador



Tapa dispositivo sensor



Carcasa dispositivo contador