UNIVERSIDAD DE CONCEPCIÓN

FACULTAD DE INGENIERÍA

DEPARTAMENTO DE INGENIERÍA ELÉCTRICA

549253 – Taller de Aplicación TIC I

Profesor: Vincenzo Caro
Ayudante: Rodrigo Hernández.

Miniproyecto 2.

Benjamín Ceballo Gil
José Figueroa
Pedro Núñez Moraga

Concepción, 24 de mayo de 2023

Resumen

Nosotros como grupo, en el presente, informamos sobre los avances realizados en este nuevo proyecto de trabajo de aplicación tecnológica como parte de la búsqueda a la necesidad de soluciones a problemas ingenieriles mediante la utilización de sistemas basados en la plataforma Raspberry Pi 3.

En este caso en particular, el desarrollo de lo que llamamos el Miniproyecto 2, en el cual nuestra misión es proveer infraestructura digital para monitoreo y control de espacios mediante un dispositivo de sensado electrónico e infraestructuras de comunicaciones por red para proveer a prospectivos clientes un sistema de fácil uso y bajo costo para necesidades de medición de condiciones de humedad y temperatura atmosférica así como también mediciones de distancia de objetos y personas que ingresen a una sala donde se instale el dispositivo, así como de también proveer el software necesario y accesible a los usuarios para realizar estos monitoreos, así como el analizar y procesar la información provista por los mismos y poder detectar posibles fallos en el sistema de manera que se den avisos preventivos para asegurar una respuesta rápida y efectiva a cualquier improvisto que pueda surgir.

Para esto presentaremos los distintos avances realizados en el transcurso del desarrollo del mismo, desde el diseño e implementación de lo que es el componente de hardware y cómo este se controla, al software empleado durante el proyecto, incluyendo también una novedad para nuestros trabajos siendo esta la integración de interfaces gráficas así como también del ensayo y puesta en marcha del sistema y la obtención de resultados del mismo.

De esta manera entonces, es que presentamos con gusto los avances realizados en la implementación de este sistema tecnológico para medición y control de condiciones de espacios cerrados de fácil uso y cómo este fue puesto en marcha en el transcurso del último tiempo que ha tomado su realización.

Tabla de Contenidos

Capítulo 1. Descripción del Proyecto:

Entendiendo la problemática:

Lo primero que debemos analizar es la necesidad particular de nuestros prospectivos clientes que les llevan a recurrir a nosotros, un equipo ingenieril de la Universidad de Concepción, como forma de buscar ayuda para la solución de una problemática real.

En particular, se nos presenta la necesidad de tener e implementar una manera de poder monitorear las condiciones internas tanto atmosféricas como espaciales de espacios cerrados y de poder analizar estadísticamente estas mediciones a modo de poder tener la capacidad de estudiar de manera más detallada la variación de las condiciones medidas para con ello sacar conclusiones sobre la seguridad del lugar.

Se debe tener también la capacidad de no solo obtener estas informaciones sino además el poder controlar el sistema en modalidad ex-situ para así facilitar de manera notoria la operación del sistema a quienes se les otorgue de modo que sean capaces no solamente de obtener la información que buscan sino también poder detectar y diagnosticar también posibles fallas en el sistema para que se puedan solucionar de manera eficiente y efectiva según sea necesario.

Propuesta de Solución:

Como equipo, hemos llegado por consenso común a una propuesta de solución que consideramos idónea para esta situación, la que consiste en la implementación de un sistema de control remoto de espacios cerrados basado en infraestructura que emplea microcomputadores Raspberry Pi puesto a la accesibilidad fácil de los mismos además de las capacidades tanto de procesamiento como de control a través de redes que entregan los mismos.

Este sistema, a nivel de hardware, consiste en una ya mencionada unidad de control la cual estará conectada a una plataforma electrónica que integra circuitería de adquisición de datos y mediciones basada en el uso de un sensor de humedad y temperatura a nuestra disposición junto a la de un instrumento de telemetría basado en un sensor que emplea emisión y recepción de ondas de ultrasonido para determinar distancias, además de una interfaz física en base a luces LED y un zumbador, los cuales permitirán a los usuarios el poder observar y diagnosticar el estado de funcionamiento del sistema para tanto verificar su adecuado funcionamiento como la recepción de

comandos y posibles errores a modo que estos puedan ser detectados a tiempo y de esta manera se pueda gestionar una necesaria mantención de manera veloz y efectiva.

Factibilidad y Materiales Necesarios:

Ahora bien, con respecto a la factibilidad de la aplicación de nuestra propuesta de solución y los materiales necesarios para la implementación y puesta en marcha de la misma, podemos observar que de buenas a primeras estamos en una buena situación respecto a lo que a factibilidad corresponde puesto a que contamos con los materiales necesarios y poseemos los conocimientos que debemos utilizar así como experiencias previas trabajando con estos materiales y disponemos de antemano con material para guiarnos y orientarnos a cómo debemos aplicar la solución que planteamos, haciendo que sea relativamente sencillo el poner en marcha al Miniproyecto 2.

Dentro de lo que es el conjunto de material de trabajo, se requiere un computador capaz de ejecutar código escrito en el lenguaje de programación Python, de los cuales disponemos y es además sencillo el poder configurar a computadores con diversos sistemas operativos para ser capaces de esto si es que no poseen las capacidades de manejar código escrito en Python de manera nativa.

Asimismo, para las funciones de control remoto se requiere de una conexión a internet tanto en el equipo que estará controlando al dispositivo como en el área donde se ponga a trabajar el dispositivo para de esta manera poder establecer una conexión segura para intercambio de información entre el computador del usuario y el dispositivo de medición.

Junto a esto, se requiere de una computadora Rasbperry Pi 3 conectada a la suite de sensores empleados, de los cuales se detalla más en el anexo adjunto al final de este documento. Estas unidades hacen interfaz con la Raspberry mediante una protoboard y el cableado correspondiente que les enlaza, por lo tanto convirtiéndose así en una solución de bajo costo y de baja dificultad de instalación y puesta en servicio.

Actividad N°1: Sistema de Control y Monitoreo Remoto

Esta primera actividad consiste en desarrollar un sistema de control y monitoreo remoto utilizando una Raspberry Pi. El objetivo principal es establecer una conexión SSH entre la Raspberry Pi y un servidor remoto para permitir el monitoreo y manipulación de los datos generados por varios sensores, permitiendo la toma de decisiones en tiempo real.

Ítem 1.1:

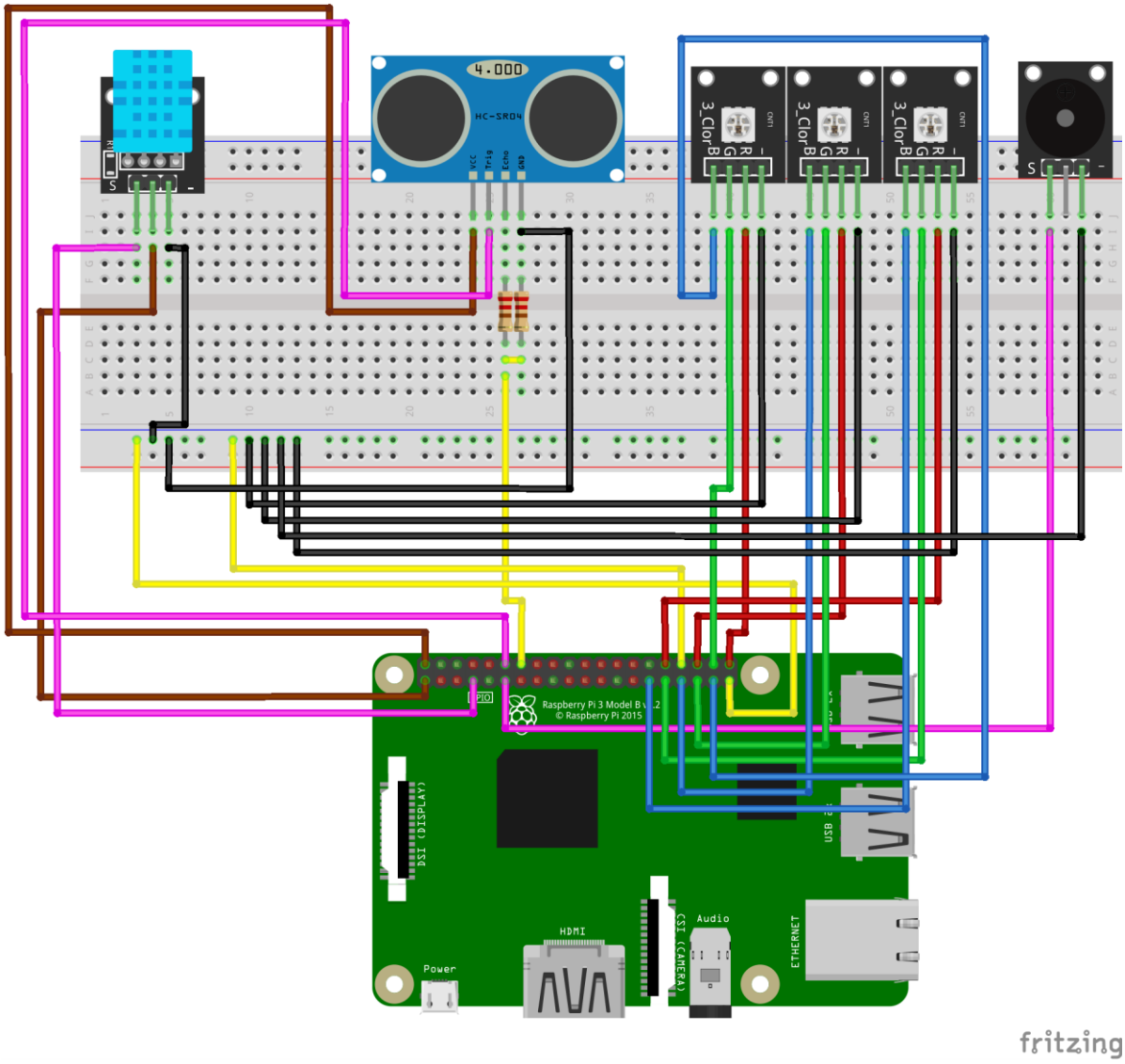
1.1a) Enunciado

Para implementar la primera parte del sistema que involucra el monitoreo remoto, consideren los siguientes módulos de su kit de 45 Sensores:

- Sensor DHT11 de temperatura y humedad (KY-015).
- Sensor HC-SR04 de distancia por ultrasonido (KY-050).

Luego, para cada sensor, preparen un código en Python desde Raspberry Pi que se ejecute de manera ininterrumpida, el cual debe leer los datos capturados (temperatura, humedad y distancia), desplegar los datos en pantalla con un máximo de 3 decimales y guardar los datos en tiempo real en un logging cada un intervalo de tiempo determinado.

1.1b) Esquemático Circuitos:



1.1c) Resultados y Comentarios:

Efectivamente, luego de realizar la ya ilustrada instalación de los sensores a ser empleados, se tomó como base el código provisto para probar su funcionalidad y a continuación este se fue modificando según resultase necesario. Se dejó en un bucle el cuerpo principal del programa a modo que este pueda ejecutarse de manera ininterrumpida haciendo también uso de los métodos *time* y *logging* para con ello proveer al programa un marco de referencia con el cual definir cada cuanto tiempo en segundos se haría cada medición primero con el sensor de humedad y temperatura que registra los datos en porcentaje y grados Celsius respectivamente midiendo cada dos segundos y asimismo a continuación el sensor de distancia, el cual está configurado para operar en un margen funcional de detección de entre dos y trescientos centímetros, marcando este cada vez que se recibe

un contacto y sobre todo a qué distancia se encuentra el objeto detectado. Estas informaciones de mediciones se van adjuntando de manera automática en un archivo de registro denominado *log.txt* en el cual se marca la fecha y hora de la medición seguida por la información de la medición realizada con los sensores.

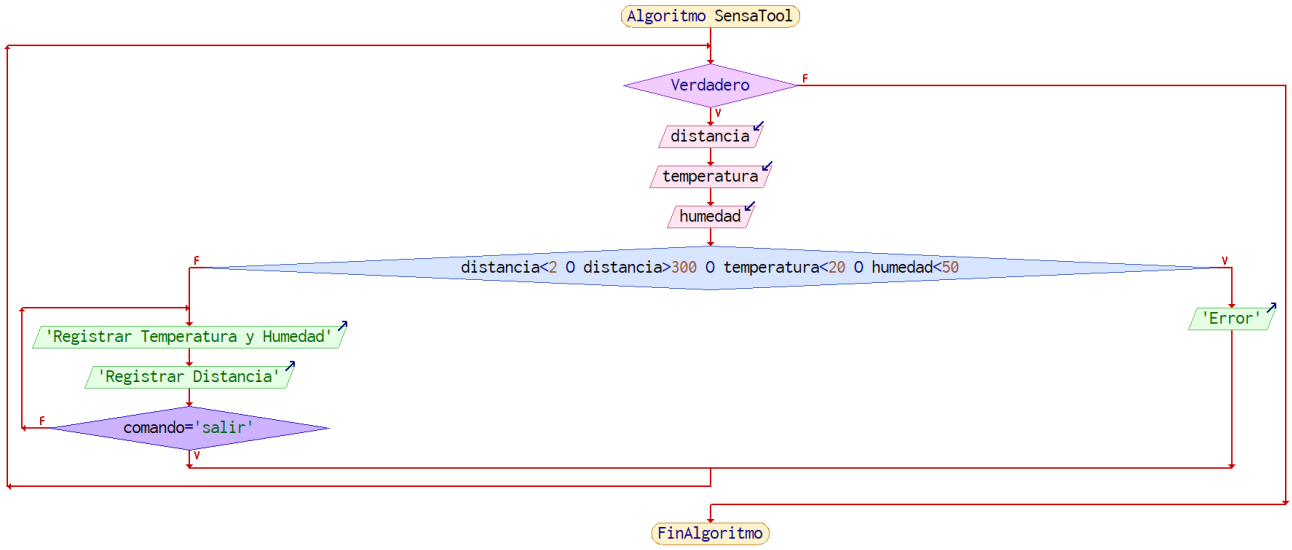


Figura 1: Esquema referencial del funcionamiento del programa.

log.txt

Archivo Editar Ver

```
2024-10-24 15:40:21,102 - INFO - Distancia Medida: 300.00 cm
2024-10-24 15:40:23,362 - INFO - Temperatura Registrada: 22.0C Humedad Registrada:49%
2024-10-24 15:40:23,362 - INFO - Distancia Medida: 300.00 cm
2024-10-24 15:40:25,623 - INFO - Temperatura Registrada: 22.0C Humedad Registrada:49%
2024-10-24 15:40:25,624 - INFO - Distancia Medida: 300.00 cm
2024-10-24 15:40:27,885 - INFO - Temperatura Registrada: 22.0C Humedad Registrada:49%
2024-10-24 15:40:27,885 - INFO - Distancia Medida: 300.00 cm
2024-10-24 15:40:30,146 - INFO - Temperatura Registrada: 22.0C Humedad Registrada:49%
2024-10-24 15:40:30,146 - INFO - Distancia Medida: 300.00 cm
2024-10-24 15:40:32,407 - INFO - Temperatura Registrada: 22.0C Humedad Registrada:49%
2024-10-24 15:40:32,407 - INFO - Distancia Medida: 300.00 cm
2024-10-24 15:40:34,668 - INFO - Temperatura Registrada: 22.0C Humedad Registrada:49%
2024-10-24 15:40:34,668 - INFO - Distancia Medida: 300.00 cm
2024-10-24 15:40:36,929 - INFO - Temperatura Registrada: 22.0C Humedad Registrada:49%
2024-10-24 15:40:36,929 - INFO - Distancia Medida: 300.00 cm
2024-10-24 15:40:39,192 - INFO - Temperatura Registrada: 22.0C Humedad Registrada:49%
2024-10-24 15:40:39,192 - INFO - Distancia Medida: 300.00 cm
```

Figura 2: Ejemplo de archivo de registro de datos de mediciones generado por el programa.

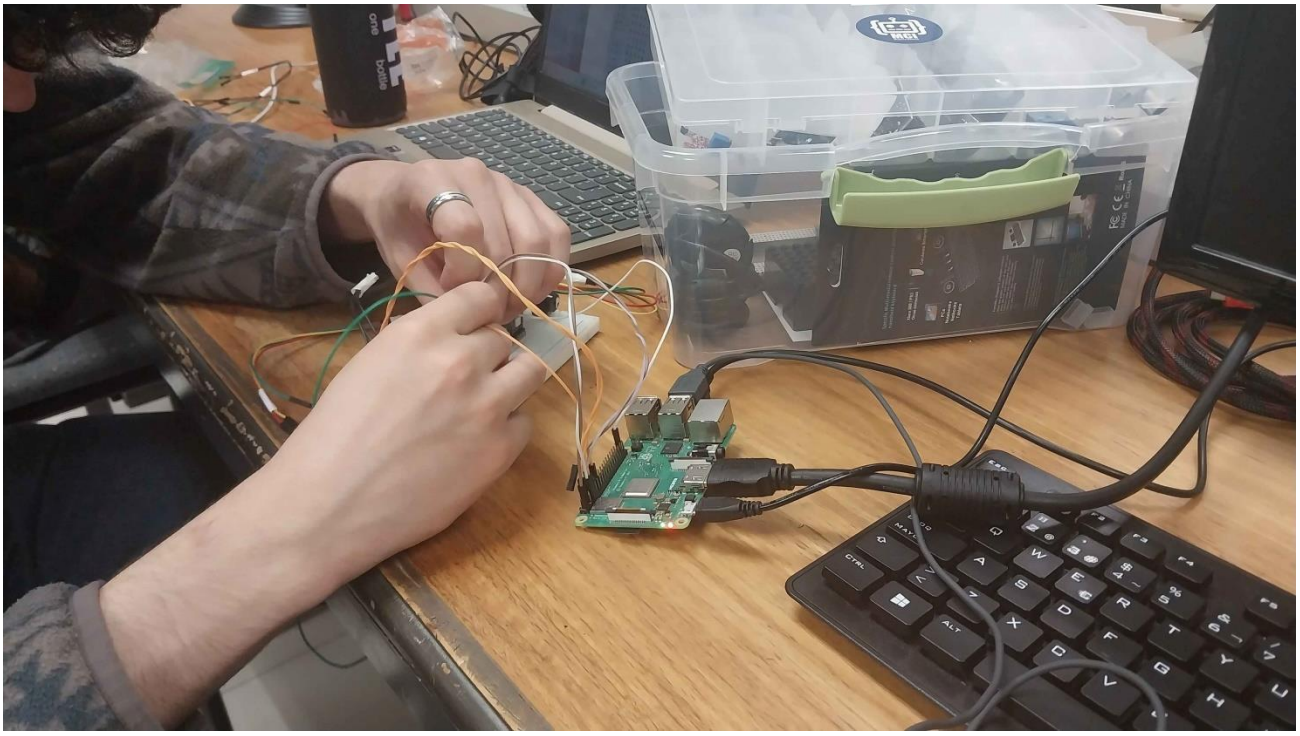


Figura 3: Instalación del Hardware.

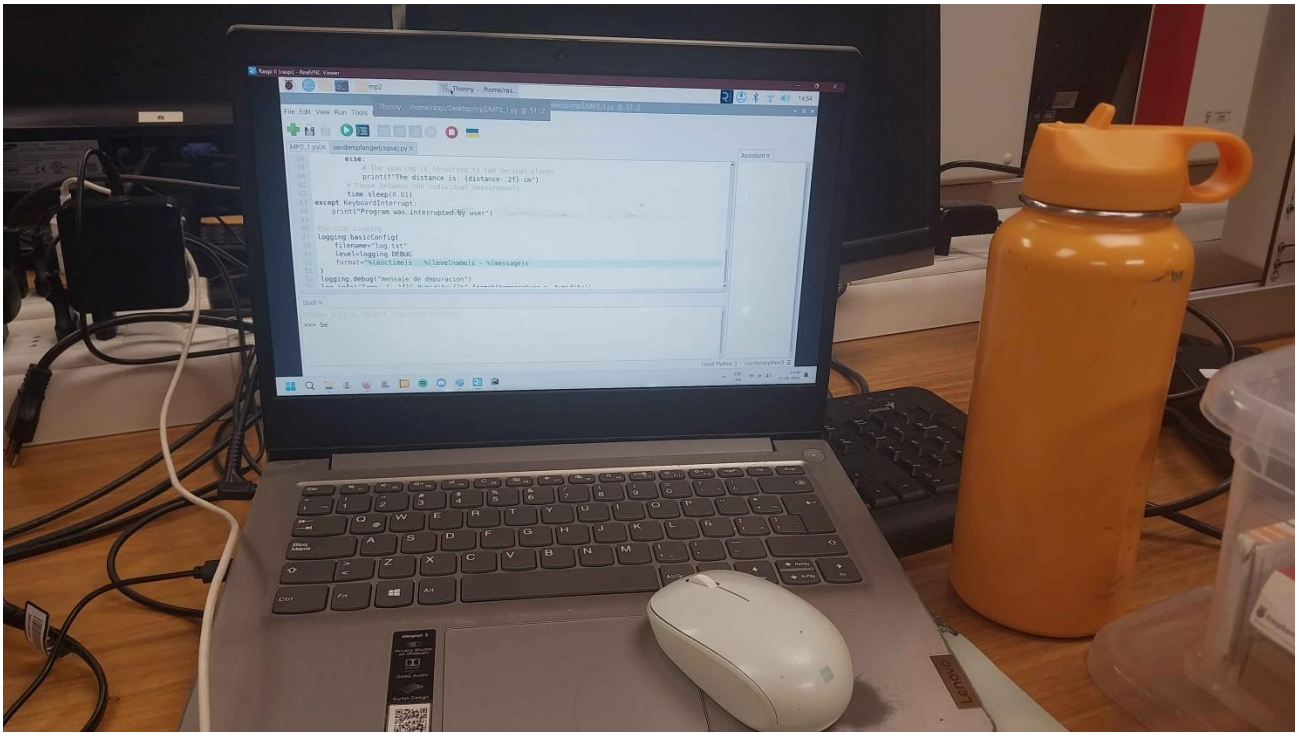


Figura 4: Escritura y prueba del código de control del disósitivo.

Ítem 1.2:

1.2a) Enunciado:

Una vez generado el archivo de logging, establezcan una conexión por SSH entre Raspberry Pi y su computador local mediante un código Python que utilice la biblioteca paramiko (desde el computador local). El código debe cumplir con los siguientes requisitos:

- Importar y mantener actualizada una copia local del archivo logging creado en Raspberry Pi en tiempo real y de manera ininterrumpida, cada un intervalo de tiempo determinado.
- Contar con un menú de selección que reciba la entrada por teclado del usuario, permitiendo elegir entre mostrar un gráfico, una tabla o terminar el proceso.
- En el caso del gráfico, este debe construirse a partir de los últimos 5 minutos de datos guardados en el logging, graficando cada variable censada con sus respectivos ejes y leyendas (por ejemplo, utilizando la biblioteca matplotlib).
- En el caso de la tabla, esta también debe construirse a partir de los últimos 5 minutos de datos guardados en el logging, mostrando por consola los valores mínimos, máximos y promedio de cada variable censada, junto con los tiempos de inicio y término registrados (por ejemplo, utilizando la biblioteca pandas)

1.2b) Resultados y Comentarios:

Respecto a lo que el trabajo de redes con Raspberry Pi para la recuperación del archivo de registro de datos de mediciones y a su vez lo que a la creación de una interfaz de línea de comandos respecta, se crearon dos programas escritos en Python para funciones específicas:

El primero corre dentro del computador que controle al sistema, en este caso empleando la biblioteca *Paramiko* la cual permite simplificar notoriamente el proceso de realizar conexiones entre dispositivos a través del protocolo *SSH* y la primera parte del código cuenta con todas las instrucciones y parámetros necesarios para asegurar una conexión segura entre el computador y la Raspberry Pi. Luego de esto tiene un bucle en el cual usando comandos de manejo de archivos entre dispositivos mediante SSH provistos por la antes mencionada biblioteca, se va recuperando de manera constante el archivo de registro *log.txt* para que este sea accesible al computador. Este programa corre sin problemas y para usarlo solo se debe modificar la carpeta de destino dentro del

computador según las preferencias del usuario así como el tener ambos equipos conectados a internet y las credenciales de acceso y conexión a la Raspberry. Se incluye también un espacio en la consola de ejecución del programa la cual permite en base a cuatro comandos escribibles por el usuario el poder terminar la ejecución del programa así como también el poder proceder a la generación de gráficas o tablas mediante la información contenida en el archivo de registro recibido.

Respecto a esto último:

Ítem 1.3:

1.3a) Enunciado:

Para implementar la segunda parte del sistema que involucra el control remoto de dispositivos, consideren los siguientes módulos de su kit de 45 Sensores:

- 3 módulos LED (KY-009, KY-011, KY-016 o KY-029).
- 1 buzzer pasivo (KY-006) o activo (KY-012).

Luego, modifiquen los códigos de los ítems anteriores para integrar las siguientes funcionalidades:

Computador local:

- Agregar una cuarta opción en el menú de selección que permita enviar alertas hacia Raspberry Pi por SSH.
- Las alertas deben activarse cada vez que el promedio de los datos de cada variable censada durante los últimos 5 minutos se escape por debajo o por sobre de un umbral a definir por ustedes.
- El envío de las alertas debe hacerse mediante un archivo (“.txt”, “.log”, “.json”, o el de su preferencia), el cual indique el estado de las tres variables al mismo tiempo

1.3 b) Resultados y Comentarios:

Ahora, para lo que fueron los ensayos de la creación de gráficos, luego de exportar el archivo de logging como un *.csv*, se logró el convertir la información en datos legibles para con ello confeccionar los gráficos deseados por el usuario de cambio de temperatura y presión atmosférica en el tiempo.

Actividad N°2: Programando una Interfaz Gráfica

Ítem 2.1:

2.1a) Enunciado:

Diseñar una interfaz gráfica en Raspberry Pi que sea capaz de interactuar con los siguientes módulos de su kit:

- 2 módulos LEDs (KY-009, KY-011, KY-016 o KY-029)

Para darle funcionalidad a su interfaz, consideren los siguientes puntos:

- Uno de los QPushButton estará encargado de encender o apagar un LED de manera digital, vinculando una única función que realice esta acción mediante el método clicked.connect().
- El QDial o QSlider estará encargado de controlar la luminosidad del segundo LED de manera análoga, vinculando una función que realice esta acción al método valueChanged.connect().
- El QLCDNumber estará encargado de mostrar en la interfaz el valor actual del QDial o del QSlider, vinculando su método display() cada vez que uno de estos widgets cambie de valor.
- Utilizando una lista o diccionario de al menos 10 imágenes con textos descriptivos de su preferencia, utilicen el primer QLabel para desplegar las imágenes en la interfaz y el segundo QLabel para desplegar el texto relacionado con estas. Luego, programen el segundo QPushButton para activar o desactivar un temporizador (QTimer) que permita que las imágenes cambien de manera aleatoria cada 5 segundos.

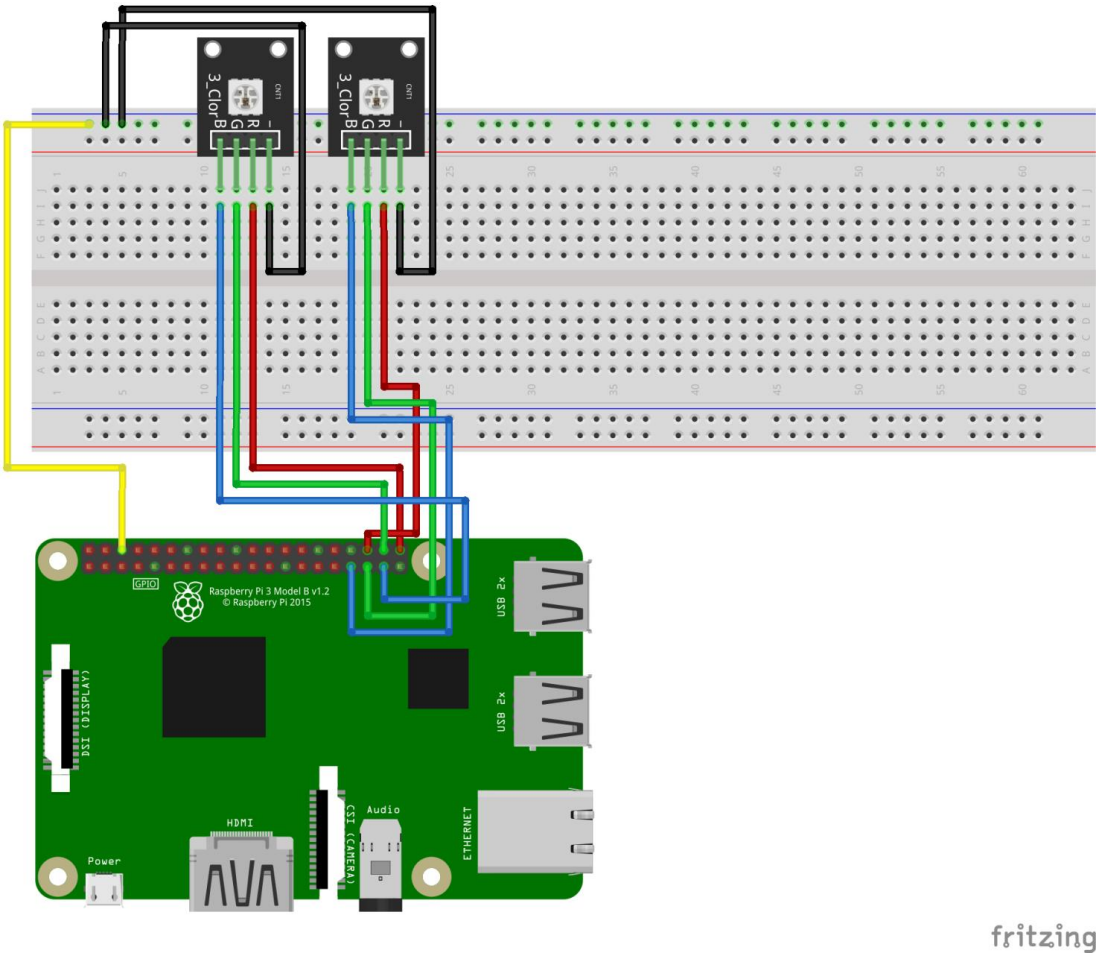
2.1a) Enunciado:

Modificar los códigos del ítem 1.2 o del ítem 1.3 para implementar una interfaz gráfica que se comunique por SSH con Raspberry Pi. Para ello, tengan en cuenta las siguientes instrucciones:

- En lugar de implementar un menú de selección por consola, implementen un menú directamente en su interfaz utilizando botones.
- En lugar de desplegar un gráfico luego de seleccionarlo en el menú, implementen un gráfico que se actualice en tiempo real, mostrando un máximo de 5 minutos de datos, utilizando para ello el backend de Qt para matplotlib u otra integración similar.
- En lugar de desplegar una tabla por consola luego de seleccionarla en el menú, implementen una tabla que se actualice en tiempo real, mostrando todas las estadísticas solicitadas anteriormente, utilizando QTableView u otra integración que opere con dataframes de pandas.

- El botón relacionado con el gráfico ahora deberá generar una imagen de este y guardarla en el dispositivo. Lo mismo para el botón relacionado con la tabla, el cual ahora deberá generar un archivo con el contenido de la tabla (por ejemplo, un “.txt” o un “.json”) y guardarla en el dispositivo.

2.1b) Esquemático Circuitos:



2.1c) Resultados y Comentarios:

2.1 c) Implementación

El front-end fue realizado en la aplicación Qt Designer, primero se realizo una interfaz básica para trabajar en darle funcionalidad y luego, con la interfaz funcionando, se perfeccionó el diseño antes mencionado. Este archivo ‘.ui’ fue transformado a uno ‘.py’ mediante la consola de anaconda. La única modificación de este archivo fue agregar 3 líneas de código, líneas 141, 142 y 143, que permiten que las imágenes y textos mostrados se adapten mejor al espacio. Este código encuentra en el archivo MascaraFinal.py (disponible en el enlace de GitHub asociado)

El back-end fue realizado en el lenguaje de programación Python, sin mayores dificultades, usando de guía principalmente el material entregado en clases y códigos realizados por chat gpt para entender el funcionamiento de ciertos comandos, y que fueron usados como base de algunas secciones de código. Este se encuentra en el archivo Interfaz.py. Primero se generó el código que hace funcionar los LEDs y, al tener esta sección lista, se hizo el código que controla el timer y los QLabels que muestran los textos e imágenes.

Los métodos asociados a los LEDs son simples, led_on( ) modifica el estado del led (encendido o apagado) y cambia la variable asociada; esta asociada a un QPushButton mediante el método "connect". La función led_lum( ) controla el segundo led a través del valor entregado por el QSlider, que se le entrega a un led para variar su intensidad; este mismo valor se muestra en la interfaz a través de QlcdNumber mediante el método "display". Para la parte de las imágenes y textos, se crearon listas con los nombres (títulos) de cada imagen, las descripciones, y la imagen misma, estas últimas entregadas con su ruta absoluta. (Se usó el comando "expanduser" de la biblioteca "os" para obtener esta ruta y así facilitar el uso del código en cualquier dispositivo, pues fue creado y probado en un computador personal y luego envido a la Raspberry). Se crearon los métodos mostrar_contenido y controlador_timer para dad funcionalidad a la interfaz. El primero selecciona un número al azar entre 0 y 9 para elegir la imagen a mostrar (con su título y descripción); este método es llamado una vez al ejecutar el código para que la interfaz comience mostrando una imagen. El segundo método permite que el QPushButton encienda y apague el timer. Por último, se relacionan el timer y el método mostrar_contenido a través del comando timer.timeout.connect (linea 40) Se logró de forma satisfactoria la implementación de la interfaz gráfica solicitada, con todas sus funcionalidades. La interfaz final utilizada como front-end es el quinto de los diseño realizados, y tiene una clara división entre la sección izquierda (leds) y derecha (imágenes-texto). La instalación del hardware asociado a la interfaz no fue de dificultad pues era simple (únicamente dos LEDs RGB).

Anexo A. Información Adicional

A.1) Sensores Utilizados:

Durante el transcurso del proyecto se emplearon los siguientes sensores para realizar las respectivas mediciones:

Sensor DHT11 de temperatura y humedad (KY-015):

Este dispositivo se conecta a la Raspberry mediante cableado dispuesto en una protoboard de modo que se le pueda otorgar alimentación eléctrica y una vía para enviar señales que comunican al sensor cuando activarse y con ello comunicarle al microcomputador la información recibida por el sensor el cual escanea el aire a su alrededor para con ello determinar la humedad la cual nosotros leemos y registramos en base a un porcentaje así como también la temperatura atmosférica del aire en el lugar donde se realicen estas mediciones a modo que este pueda ser registrado, en nuestro caso en grados centígrados para su posterior análisis.

Sensor HC-SR04 de distancia por ultrasonido (KY-050).

Este dispositivo se instala y opera de similar manera al antes mencionado, pero en este caso consta de un sistema de operación diferente pues es un sonar que emplea ultrasonido para detectar objetos sólidos a la distancia frente a sí con una bocina emisora la cual envía la onda la cual es recibida por una receptora ubicada al lado una vez esta logre tocar una superficie sólida en su camino y se refleje en la misma, permitiendo determinar distancias gracias a los desplazamientos observables por el efecto Doppler. Tiene un rango máximo de 3 metros y se ha configurado para que si las medidas están sobre este margen o bajo el margen de dos centímetros, el programa controlador reporte un error y estos datos son almacenados en centímetros.

A.1.1 Paramiko y SSH:

Para todas las labores de interfaz entre dispositivos mediante redes inalámbricas optamos por hacer las conexiones mediante el protocolo *Secure Shell* mejor conocido como *SSH* el cual consiste en un protocolo para establecer conexiones de manera remota entre dos dispositivos donde uno actúa como cliente y el otro de servidor mediante la dirección IP y una contraseña de acceso que actúan como credenciales para poder acceder de manera remota en este caso a la Raspberry Pi desde un PC y también al PC desde la Raspberry para lo que ha sido el envío del archivo de registro de

datos logging.txt así como también de las advertencias para regular el sistema de monitoreo de estado del dispositivo respectivamente. Esto se ha hecho utilizando la red del laboratorio de señales donde se realizan las sesiones de trabajo así como también se puede hacer con redes inalámbricas externas a este recinto siempre y cuando se cuenten con los datos necesarios como lo son la dirección IP, la clave de acceso y por supuesto el canal de comunicaciones.

Para la implementación del código necesario para hacer funcionar nuestros programas hemos utilizado la biblioteca *Paramiko* la cual nos permite programar en Python los comandos necesarios para gestionar las operaciones de conexión, acceso y transferencia de archivos mediante el protocolo SSH de manera sencilla y rápida para así agilizar el funcionamiento de nuestro programa y la puesta en marcha de nuestro dispositivo.

A.1.2 Herramientas para interfaces gráficas:

Para la implementación de interfaces gráficas hemos estado utilizando la biblioteca *PyQT*, la cual es una de las más populares y con mayor amplitud de herramientas a nuestra disposición para la creación y ejecución interfaces gráficas para controlar distintos componentes de algún software. Primeramente se comenzó ensayando el diseño de la misma utilizando el programa *QT Designer* el cual permite esbozar una maqueta de la interfaz la cual luego se exporta y codifica para que pueda funcionar usando un IDE en Python de preferencia del usuario con las herramientas, funciones y métodos que brinda la ya mencionada biblioteca de código.