TECNOLÓGICO NACIONAL DE MÉXICO

INSTITUTO TECNOLÓGICO DE TUXTLA GUTIÉRREZ

DEPARTAMENTO DE ELECTRICA Y ELECTRONICA

INGENIERÍA ELECTRÓNICA

REPORTE TECNICO DE RESIDENCIA PROFESIONAL

SISTEMA PARA MONITOREO Y CONTROL DE CULTIVOS VERTICALES.

Autor: Martín Chávez Morales

No. De control: 14270560

Asesor interno: M. en C. José A. Zepeda Hernández

TUXTLA GUTIERREZ, CHIAPAS, AGOSTO-DICIEMBRE 2018

INDICE

CAPITULO I. 1
INTRODUCCIÓN. 1
Antecedentes. 1
Estado del arte. 6
Justificación. 8
Objetivos. 9
Objetivo general. 9
Objetivos específicos. 9
CAPITULO II. 10
Caracterización del área en que se participó. 10
Problemas a resolver, priorizándolos 11
Alcances y limitaciones. 12
CAPITULO III. 13
Fundamento teórico. 13
Sistema microcontrolador. 13
Ethernet Shield. 16
Higrómetro fc-28. 19
Modulo sensor de intensidad de luz LDR. 21
Sensor de temperatura y humedad DHT22 o AM2302. 22
Protocolo MQTT: (Message Queue Telemetry Transport). 25
Desarrollo. 29
Programación de los sensores en Arduino. 29
Creación del servidor que alojara los datos. 43
Configuración entre Servidor y Arduino. 48
Configuración de un cliente “Android”. 54

Diseño de la fuente de distribución de voltaje. 59
Diagrama de conexiones..... 64
Distribución de los sensores en el jardín vertical. 66
CAPITULO IV..... 67
Resultados y conclusiones. 67
Resultados. 67
Conclusiones..... 75
Bibliografía..... 76

INDICE DE IMAGENES

Imagen 1 Arduino Mega 2560. 13
Imagen 2 Arduino Ethernet Shield R3. 16
Imagen 3 HIGRÓMETRO FC-28. 19
Imagen 4 ESQUEMA DE MONTAJE. 20
Imagen 5 Descripción de Higrómetro. 20
Imagen 6 sensor de presión BMP180. 21
Imagen 7 Conexiones entre Arduino y módulo BMP180. 22
Imagen 8 Pines de DHT 11 / DHT 22. 23
Imagen 9 Conexiones entre Arduino y el sensor DHT22 24
Imagen 10 El esquema eléctrico queda como la siguiente imagen. 24
Imagen 11 Topología MQTT. 25
Imagen 12 Jerarquía MQTT. 26
Imagen 13 modelo de publicación y suscripción MQTT para sensores IoT. . 28
Imagen 14 ingreso a la página web. 29
Imagen 15 Inicio de sesión. 29
Imagen 16 Creación de un nuevo proyecto. 30
Imagen 17 Selección del tipo de proyecto a crear. 30
Imagen 18 Elección del tipo de placa con la que se trabajara. 31
Imagen 19 Definición del nombre del proyecto. 31
Imagen 20 Ventana principal del proyecto. 32
Imagen 21 Sección MQTT. 32
Imagen 22 Inicializar el programa. 33
Imagen 23 Creación de un ciclo. 33
Imagen 24 Determinación de tiempo del ciclo. 34
Imagen 25 Bloque para enviar datos al servidor. 34
Imagen 26 Ingreso al ciclo. 35
Imagen 27 Sensor DHT-22. 35
Imagen 28 Instrucción para leer la temperatura. 36
Imagen 29 Duplicar instrucción. 36
Imagen 30 Lectura de la humedad relativa. 37
Imagen 31 Instrucción de lectura de humedad. 37
Imagen 32 Definición de la instrucción de lectura de humedad. 38
Imagen 33 Segunda instrucción de lectura de humedad. 38
Imagen 34 Tercera instrucción para lectura de humedad del suelo. 39
Imagen 35 Sensor LDR. 39
Imagen 36 Creación de la instrucción para lectura de iluminación. 40
Imagen 37 Instrucciones para lectura de iluminación. 40

Imagen 38 Inicio de ArduinoBlocks-Connector. 41
Imagen 39 ArduinoBlocks-Connector corriendo y listo para la conexion. 41
Imagen 40 Detección del puerto donde se conectó Arduino. 42
Imagen 41 Subida del código a la placa. 42
Imagen 42 CloudMQTT. 43
Imagen 43 Ventana principal del servidor. 43
Imagen 44 Definición de la instancia. 44
Imagen 45 Definir región. 44
Imagen 46 Selección de la región. 45
Imagen 47 Confirmación de creación. 45
Imagen 48 Instancia creada en el servidor. 46
Imagen 49 Detalles del servidor 46
Imagen 50 Creación de usuario en el servidor 47
Imagen 51 Creación de ACLs. 47
Imagen 52 Proyecto en ArduinoBlocks.ç 48
Imagen 53 Dirección del servidor 48
Imagen 54 Definición de la dirección del servidor en Arduino. 49
Imagen 55 Puerto de conexión. 49
Imagen 56 Definición del puerto de conexión. 50
Imagen 57 Usuario del servidor. 50
Imagen 58 Ingreso del nombre de usuario para el acceso desde Arduino. 51
Imagen 59 Contraseña de acceso al servidor. 51
Imagen 60 Ingreso de la contraseña. 52
Imagen 61 Arduino IDE. 52
Imagen 62 Continuación del código en Arduino IDE. 53
Imagen 63 App MQTT. 54
Imagen 64 Ventana principal de la App. 54
Imagen 65 Definición de los parámetros de conexión. 55
Imagen 66 Ventana principal del servidor visto desde Android. 55
Imagen 67 Suscripción a Humedad. 56
Imagen 68 Suscripción a Temperatura. 56
Imagen 69 Suscripción a Tierra 1. 57
Imagen 70 Vista de las suscripciones. 57
Imagen 71 Vista de todas las suscripciones. 58
Imagen 72 Diseño en Proteus. 59
Imagen 73 Vista PCB Layout. 59
Imagen 74 3D Visualizer. 60
Imagen 75 Diseño en PDF. 60
Imagen 76 Planchado de la placa. 61

Imagen 77 Placa introducida a Cloruro Férrico. 61
Imagen 78 Placa con recubrimiento de tóner. 62
Imagen 79 Placa perforada. 62
Imagen 80 Componentes en la placa. 63
Imagen 81 Componentes soldados. 63
Imagen 82 Fuente terminada. 63
Imagen 83 Conexión de Arduino. 67
Imagen 84 Placa de distribución. 67
Imagen 85 Sondas de humedad. 68
Imagen 86 Lecturas recibidas por el servidor. 68
Imagen 87 Ventana de lecturas recibidas por el servidor. 69
Imagen 88 Ventana principal App MQTT. 69
Imagen 89 Vista de las lecturas. 70
Imagen 90 Lecturas de temperatura. 71
Imagen 91 Lecturas de humedad. 71
Imagen 92 Lecturas de iluminación del sensor LDR 1. 72
Imagen 93 Lecturas de iluminación del sensor LDR 2. 72
Imagen 94 Lecturas de iluminación del sensor LDR 3. 73
Imagen 95 Lecturas de humedad del suelo de la sonda 1. 73
Imagen 96 Lecturas de humedad del suelo de la sonda 2. 74
Imagen 97 Lecturas de humedad del suelo de la sonda 3. 74

CAPITULO I.

INTRODUCCIÓN.

Antecedentes.

En el año 2017 el Tecnológico Nacional de México emitió la convocatoria para que diferentes Institutos del país pudieran incorporarse al Proyecto “4X Metro Centro de Innovación Industrial”, que es un consorcio integrado por el propio Tecnológico Nacional y tres asociaciones de Clusters del país cuyo objetivo es fomentar la adopción de tecnologías emergentes, en especial las relacionadas con Industria 4.0. El estado de Chiapas tuvo una participación muy destacada ya que uno de los cuatro integrantes del Consorcio es el Cluster de Tecnologías de la Información de Chiapas A.C., el cual ya tenía varios proyectos a desarrollar en el tema de Industria 4.0. El Instituto Tecnológico de Tuxtla Gutiérrez atendió a la convocatoria y fue seleccionado para participar en dicho Proyecto. Las actividades en el año 2018 iniciaron con la impartición de los cursos “Estrategias de innovación y modelos diferenciados de negocio para la Industria 4.0” y “Arquitectura de Tecnologías 4.0” impartidos en los meses de enero y febrero. En el primer curso se identificaron ideas de proyectos que podrían ser susceptibles de desarrollarse con empresas locales orientadas hacia el sector de agroindustria y se convocaron a empresarios del ramo para presentarles el proyecto.

Paisajismo y Riegos GreenBox S.P.R. de R.L. de C.V., es una empresa ubicada en Tuxtla Gutiérrez que se dedica, al diseño y desarrollo de áreas verdes, azoteas verdes, huertos urbanos, jardines verticales y al mantenimiento de los mismos. (www.greenbox.com.mx). Esta empresa respondió a la convocatoria del Instituto Tecnológico de Tuxtla Gutiérrez y se presentó al primer curso identificándose con mucho interés para participar en el proyecto de Industria 4.0. En la aplicación de la técnica de Design Thinking, se identificó que una de las principales debilidades que presentan es el seguimiento del mantenimiento de las instalaciones que atienden. En su programa de mantenimiento ordinario deben realizar visitas programadas a todas las instalaciones que han realizado y siguen atendiendo, para que los diferentes cultivos vegetales se preserven con las condiciones adecuadas para su correcto desarrollo. Esta actividad les representa una gran inversión de tiempo y recursos ya que la inspección se realiza de forma visual y en cada visita tienen que revisar las condiciones ambientales en que se encuentran las plantas. Sin embargo, en ocasiones han surgido imprevistos ocasionados por

la inadecuada operación de los propietarios de las instalaciones e incluso han llegado a perderse algunas de estos desarrollos. Lo anterior bien podría evitarse si contaran con un sistema de alerta que les permitiera conocer la situación de los cultivos antes de que se presentaran situaciones irreversibles para las plantas.

En el desarrollo del curso “Arquitectura de Tecnologías 4.0”, se identificó a la Industria 4.0 como la aplicación colectiva de las tecnologías y conceptos a lo largo de la cadena de valor de una empresa. En ella se conjugan aspectos de control de sistemas físicos mediante programación, creando una copia virtual del mundo físico y permite tomar decisiones basadas en la organización de estos mecanismos. Asimismo, este concepto toma en cuenta el incremento de la digitalización en la industria para que los objetos físicos sean integrados con información de la red, permitiendo la descentralización de la producción y su adaptación en tiempo real. Lo anterior nos lleva a los principios de diseño que se analizarán y aplicarán para lograr mejorar la situación del mantenimiento de los jardines verticales que la empresa GreenBox tiene para con sus clientes.

El presente proyecto se encuentra alineado con el PIID del Instituto Tecnológico de Tuxtla Gutiérrez en los objetivos, estrategias y líneas de acción siguientes: Objetivo 4. Impulsar la ciencia, la tecnología y la innovación en el Instituto Tecnológico de Tuxtla Gutiérrez; Estrategia 4.1. Impulsar el desarrollo de las capacidades científicas y tecnológicas con enfoque en la vocación productiva de las regiones de la zona de influencia del Instituto y áreas aledañas; Línea de acción 4.1.5. Alinear las acciones de generación y aplicación innovadora del conocimiento con las necesidades de los sectores estratégicos. Objetivo 5. Consolidar la vinculación del Instituto Tecnológico de Tuxtla Gutiérrez con los sectores Público, Social y Privado; Estrategia 5.3. Impulsar la transferencia de conocimiento y de desarrollo tecnológico al sector productivo; Línea de acción 5.3.4. Gestionar y generar proyectos que respondan a las necesidades de desarrollo tecnológico que involucren la inversión pública y privada.

Actualmente diversos sistemas y procesos se encuentran automatizados, es decir, se han aplicado sistemas mecánicos y electrónicos para controlar el proceso y no requerir intervención humana para su funcionamiento. La automatización se ha desarrollado rápidamente, se han creado nuevos avances tecnológicos y técnicas de control, que son aplicados en diversas áreas como la industria, medicina, aeronáutica, entre otros. En el área de la

agronomía, la automatización cumple un rol muy importante debido a que optimiza los procesos, incrementando la productividad y mejora la calidad de los productos, esto ayuda a satisfacer las exigencias del mercado. Por otro lado, los avances dentro del control climático han permitido la mejora de sistemas como es el caso de los invernaderos. Diferentes métodos y modelos de control son aplicados para controlar el comportamiento de las variables climáticas que afectan el entorno del invernadero. Se usan desde sistemas de control usando lógica clásica hasta sistemas de control inteligente, que usan lógica difusa, redes neuronales, entre otros.

La agricultura vertical busca asegurar la sustentabilidad atendiendo la seguridad alimentaria de la creciente población de los centros urbanos. En principio, es un concepto simple, cultivar hacia arriba en vez de hacerlo horizontalmente, (Despommier, 2010). Los cultivos verticales son estructuras con varios niveles de camas de crecimiento y se instalan en espacios que brinden un ambiente apropiado al crecimiento vertical de las plantas. Los cultivos verticales aprovechan las mismas tecnologías que han hecho posible el desarrollo de los invernaderos de alta tecnología, siendo por ello un tipo de agricultura de ambiente controlado (AAC) (Despommier, 2014).

A partir de los años noventa, con el desarrollo de sistemas hidropónicos y aeropónicos sofisticados, la agricultura de ambiente controlado ha madurado hacia una estrategia viable comercialmente para la producción a gran escala de una amplia variedad de cultivos. La hidroponía es la ciencia de hacer crecer a las plantas en ausencia de suelo. Típicamente emplea un sistema de crecimiento tubular para sostener a las plantas, que proporciona una delgada película de nutrientes disueltos a las raíces que se encuentran en contacto con la fase líquida del sistema (Matuszak, 2012). Los invernaderos hidropónicos se han convertido en una práctica comercial ya que ofrecen muchas ventajas sobre las prácticas de cultivo tradicional. En algunos casos, los cultivos con tecnologías hidropónicas pueden generar hasta ocho cosechas en el año, mientras que aquellos en condiciones tradicionales pueden generar solamente hasta dos o tres cosechas anuales. Las pérdidas en cultivos hidropónicos oscilan alrededor del diez por ciento, mientras que en la mayoría de los casos empleando equipos y métodos actualizados, desde la plantación hasta la cosecha, en países en desarrollo las pérdidas exceden al setenta por ciento (Carruthers, 2007).

Se han logrado importantes avances en el desarrollo de métodos de cultivos verticales con la idea de lograr la producción sustentable de alimentos. El diseño, distribución y configuración de estos cultivos de alta tecnología proporcionan una exposición óptima de iluminación con una precisa dosificación de nutrientes para cada planta. Diseñados para proporcionar un crecimiento en un ambiente controlado, estos cultivos eliminarían la necesidad de aplicar herbicidas y plaguicidas, maximizando la nutrición y el valor de los alimentos en el proceso (Hedenblad, 2017)

Las investigaciones se orientan hacia el desarrollo, perfeccionamiento y adaptación de estos sistemas de manera que puedan ser replicados por todo el mundo proporcionando los mejores rendimientos de producción y mínimos impactos ambientales. Estos sistemas representan un cambio de paradigma en el cultivo y producción de alimentos, siendo también adecuados para cultivos urbanos donde la disponibilidad de suelo es limitada (Al-Kodmani, 2018).

El término “Industria 4.0” llegó a ser conocido públicamente en 2011, cuando una iniciativa llamada “Industrie 4.0” –una asociación de hombres de negocios, políticos y académicos- promovieron la idea como un acercamiento al fortalecimiento de la competitividad de la industria de manufactura Alemana (Schawb, 2016). El Gobierno federal Alemán apoyó la idea anunciando que Industrie 4.0 sería una parte integral de su iniciativa “Estrategia de Alta-Tecnología 2020 para Alemania” apuntando hacia el liderazgo de la innovación tecnológica. El “Grupo de trabajo Industrie 4.0”, que se formó posteriormente, desarrollaron las primeras recomendaciones para la implementación y se publicaron en Abril de 2013 (Kagermann et al. 2013). La visión de esta iniciativa señala que en el futuro, las empresas establecerán redes globales a las que incorporaran su maquinaria, sistemas de almacenamiento en instalaciones de producción en Sistemas Ciber-Físicos (SCF). En el ambiente de manufactura, estos sistemas ciber-físicos comprenderán maquinas, sistemas de almacenamiento e instalaciones de producción inteligentes capaces de intercambiar información de forma autónoma, iniciando acciones y controlándose de forma independiente. Esto facilitará mejoras sustanciales a los procesos industriales involucrados en manufactura, ingeniería, uso de materiales y administración de la cadena de suministro y del ciclo de vida. Las Fábricas Inteligentes, que ya empezaron a funcionar, parecen emplear un sistema de producción completamente diferente. Los productos inteligentes se identifican unívocamente, pueden ser ubicados todo el tiempo y saber su propia historia, estado actual y rutas alternativas para alcanzar su estado

objetivo final. Los sistemas de manufactura embebidos están conectados verticalmente con procesos de negocios dentro de las fábricas y empresas y conectados horizontalmente a redes de valor descentralizadas que pueden ser administradas en tiempo real-desde el momento en que es colocada una orden hasta la logística externa. Además, estas dos últimas habilitan y requieren ingeniería de punto a punto a lo largo de la cadena de valor (Hermann et al. 2015).

Las recomendaciones para la implementación del “modelo Industrie 4.0” parte del análisis de los componentes de su visión. Se identifican cuatro componentes clave: Sistemas ciber-físicos, Internet de la cosas, Internet de los servicios y Fábrica inteligente. La comunicación máquina a máquina (M2M) y los Productos inteligentes no se consideran componentes independientes. M2M es un habilitador del Internet de las cosas mientras que los Productos inteligentes son un subcomponente de los Sistemas ciber-físicos. De igual forma, el Big Data y la Computación en la nube se consideran servicios de datos que utilizan los datos generados en las implementaciones de los otros componentes (Kagermann, 2013).

De la visión y los componentes de Industrie 4.0, se deriva la definición operativa que se empleará en el presente proyecto. La Industrie 4.0 es un término colectivo para las tecnologías y conceptos de organización de la cadena de valor. Dentro de la estructura modular de las Fábricas inteligentes de Industrie 4.0, los sistemas ciber-físicos monitorean los procesos físicos, crean una copia virtual del mundo físico y toman decisiones descentralizadas. Los servicios internos y externos se ofrecen sobre el Internet de las cosas y son utilizados por los participantes de la cadena de valor. Con la visión, los componentes y la definición anteriores, se derivan los principios de diseño para los escenarios de Industrie 4.0. Estos principios sirven de orientación a las empresas para identificar las áreas de oportunidad que luego pueden implementarse. En total, se identifican seis principios: 1) Interoperabilidad, 2) Virtualización, 3) Descentralización, 4) Capacidad de tiempo real, 5) Orientación al servicio y 6) Modularidad (Hermann et al. 2015). Con todas estas herramientas y sus definiciones operativas se desarrollará el prototipo de Industria 4.0 para el monitoreo y control de cultivos verticales para la empresa GreenBox.

Estado del arte.

La producción de alimentos se encuentra cada día más amenazada por condiciones inusuales de clima, escasez de agua e insuficiencia de tierras. Se estima que la población mundial pasará de siete mil millones en 2011 a nueve mil trecientos millones en el año 2050, y la población urbana pasará de tres mil seiscientos millones a seis mil trescientos millones, un incremento de setenta y dos por ciento. Debido a las limitaciones de los recursos naturales, el noventa por ciento del crecimiento en la producción global de cultivos se espera conseguir con mayores rendimientos e intensidad en los cultivos, y el restante diez por ciento de la expansión de la tierra productiva cultivable (FAO, 2009)

En el año 2010, la población de México fue de 114.3 millones de habitantes, 77.8% de los cuales se encontraban en áreas urbanas. En México, se consideran áreas urbanas aquellas con una población mayor de 2,500 habitantes (INEGI_A, 2018). De acuerdo con estimaciones recientes, la población de México alcanzará los 150.8 millones de habitantes en el año 2050 (INEGI_B,2018), (CONAPO, 2018). En consecuencia, el país requerirá más tierras para producir alimentos del sector primario, lo que a su vez requerirá que sean transportados a las áreas urbanas. Para atender las necesidades de alimentos proyectadas, se requerirá mayor cantidad de tierras cultivables, conduciendo a cambios en el uso de suelo y la cubierta vegetal, mayor deforestación y mayor presión hacia los recursos naturales y ecosistemas. La disponibilidad de los recursos hídricos sigue una tendencia similar, hay agua suficiente pero distribuida de forma desigual. Con la finalidad de alimentar a la población, proteger al ambiente, mejorar la salud humana y conseguir el crecimiento económico, se requiere una nueva forma de agricultura.

Dentro de las alternativas que han evolucionado en los últimos años, la tecnología del cultivo hidropónico ha comenzado a destacar en los países en desarrollo debido a que el crecimiento poblacional en las zonas urbanas ha representado una oportunidad para cultivar alimentos cerca de los consumidores (Lakkireddy et al., 2012). Los 100-200 millones de productores urbanos en todo el mundo que abastecen a los mercados de las ciudades con productos agrícolas frescos son evidencia de cómo puede garantizarse la seguridad alimentaria mediante la agricultura urbana (Orsini et al., 2013). En el mismo sentido, ha surgido el concepto de “agricultura vertical”.

La agricultura vertical, como un componente de la agricultura urbana, es la práctica de producir alimentos en capas apiladas verticalmente, superficies

inclinadas verticalmente y/o integradas en otras estructuras (Despommier, 2009). En 1915, Gilbert Ellis Bailey acuñó el término “vertical farming”, (que en español se usa como sinónimo de agricultura vertical, granjas verticales o bien jardines verticales) y desde entonces científicos y arquitectos han puesto su atención en la idea de producir alimentos en ambientes urbanos debido al constante crecimiento poblacional y las presiones ejercidas sobre los recursos para la producción de alimentos. Actualmente, una agricultura urbana más evolucionada, donde el producto se desarrolla en un ambiente urbano totalmente controlado dentro de estructuras verticales, ha llamado la atención en varios países. La tecnología de la agricultura vertical aplicada en espacios físicos urbanos se ha visualizado como la solución a los problemas de limitación de tierras apropiadas para la agricultura, así como también a un uso más racional del agua, dando mejores oportunidades para un abastecimiento sustentable de alimentos tanto en países en desarrollo como países desarrollados (Besthorn, 2013). Debido a los avances en las tecnologías hidropónica y aeropónica, luz mediante LEDs y energía proveída con luz solar, ya es posible tener una agricultura en las áreas urbanas y posiblemente hasta en casas habitación para crear centros de producción y consumo integrados con comunidades urbanas y suburbanas, no obstante, se hace evidente la necesidad de contar con mayores controles para su adecuado desarrollo (Despommier, 2009).

Estos controles pueden ser proporcionados por los desarrollos de la industria 4.0. La cual se define como una nueva forma de organización y funcionamiento de la industria, en la cual se implementa una conexión que va desde el usuario hasta la fabricación de un producto. A la industria 4.0 se le considera como la cuarta revolución industrial, en la que las formas de producción hacen uso de sistemas físicos y cibernéticos para crear una industria más flexible y de carácter reconfigurable, permitiendo así que la estructura de una fábrica se pueda modificar para poder producir diferentes productos (Sosa Cruz, 2017).

La finalidad del presente proyecto es precisamente la aplicación de los conceptos de Industria 4.0 mediante el desarrollo de un prototipo de Industria 4.0 para el monitoreo, seguimiento y evaluación de cultivos verticales para la empresa GreenBox S. de P.R. de R.L.).

Justificación.

La empresa GreenBox Sociedad de Producción Rural de Responsabilidad Limitada y Capital Variable (S.P.R.L. y C.V.), ha tenido un crecimiento importante en los últimos cinco años en el estado de Chiapas. Su giro es el de diseño, desarrollo y mantenimiento de cultivos verticales para empresas y dependencias de gobierno. Dado que el mantenimiento implica un seguimiento puntual de las características de operación de los cultivos en desarrollo, surge la necesidad de contar con un mecanismo de monitoreo integral que le permita a la empresa atender contingencias, pero también tener información actualizada de la operación de los todos los cultivos a los que da mantenimiento. El presente proyecto propone el diseño y desarrollo de un prototipo que le permita a la empresa GreenBox hacer este procedimiento empleando las tecnologías de la Industria 4.0.

Realizando todos los diseños y pruebas del prototipo dentro de las instalaciones del Instituto Tecnológico de Tuxtla Gutiérrez, en el laboratorio de industrias 4.0 que se ubica dentro del Edifico I del departamento de ingeniería eléctrica y electrónica.

Objetivos.

Objetivo general.

Aplicar los principios de diseño y desarrollo bajo la filosofía de Industria 4.0 para el desarrollo de un prototipo que coadyuve en la operación del mantenimiento de los cultivos verticales diseñados y atendidos por la empresa GreenBox en la ciudad de Tuxtla Gutiérrez.

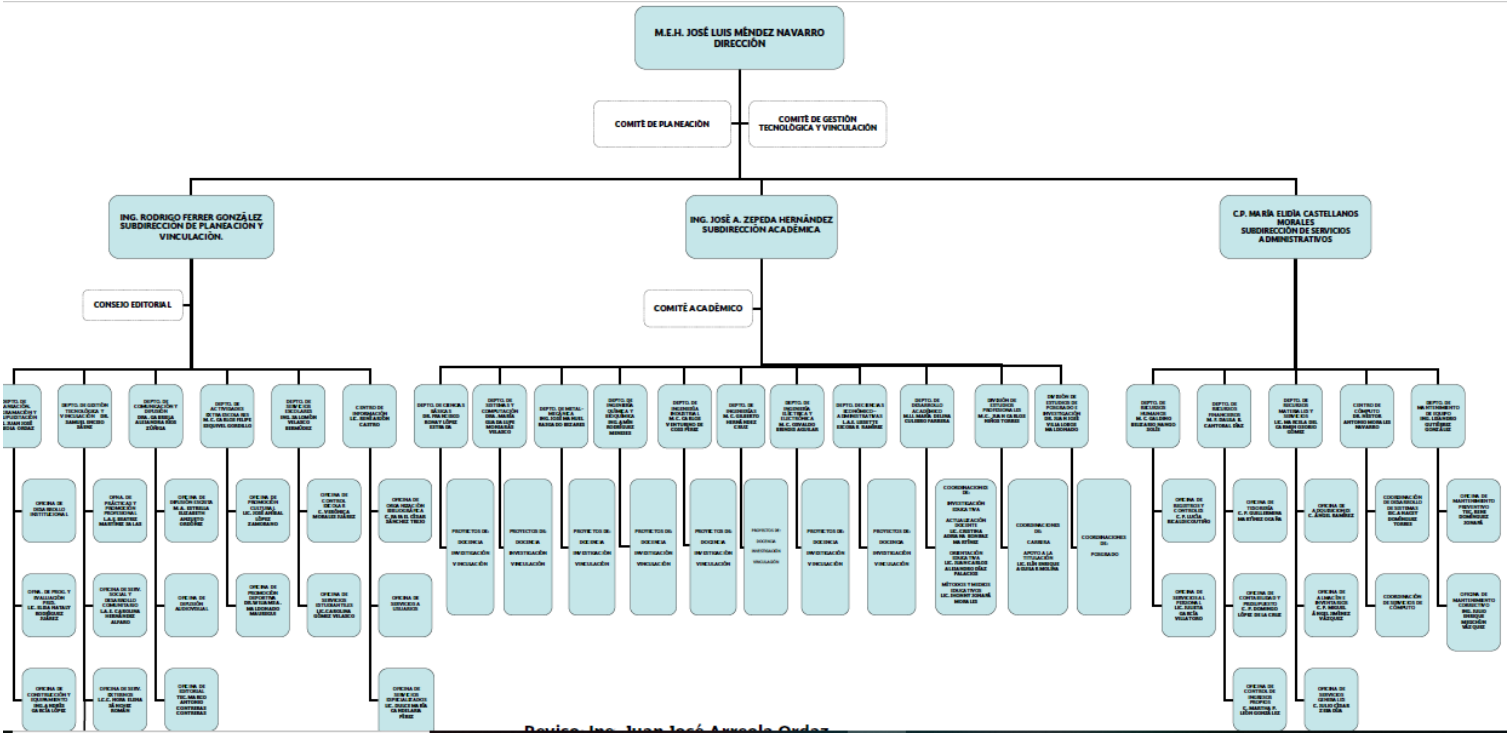
Objetivos específicos.

- Realizar el diagnostico de las necesidades de supervisión de los principios indicadores de operación de un jardín vertical prototipo de la empresa GreenBox.
- Seleccionar las tecnologías apropiadas de Industria 4.0 para la supervisión de jardines verticales.
- Diseñar y desarrollar el prototipo de Industria 4.0 para la supervisión de jardines verticales.
- Diseñar y generar el software de Internet de las Cosas (IoT) para el funcionamiento integral del prototipo.
- Realizar las pruebas de operación del prototipo con jardines verticales en operación.
- Evaluar la operación del prototipo en términos de la eficiencia de los recursos en operación.
- Realizar los informes de resultados correspondientes.

CAPITULO II.

Caracterización del área en que se participó.

Organigrama de la empresa.



Realizando el proyecto en el departamento de ingeniería eléctrica y electrónica, a cargo de M. en C. Osvaldo Brindis Velázquez. Dicho departamento se encuentra en lo que es el Edificio I, siendo en el laboratorio de industrias 4.0 de dicho edifico.

Problemas a resolver, priorizándolos.

El principal problema es que la empresa actualmente atiende en el Estado de Chiapas no cuenta con el seguimiento y control de la operación de las diferentes instalaciones de cultivos verticales que tiene en el Estado de Chiapas. No obstante, debido a la alta demanda que la empresa ha tenido, se han presentado contingencias de atención que han conducido a la pérdida de las plantas cultivadas con las consecuentes mermas tanto económicas como de confiabilidad en el servicio de la empresa.

La empresa reconoció el potencial de las tecnologías de la Industria 4.0 en su propio desarrollo en una de sus líneas de trabajo, que es el diseño, construcción y operación de cultivos verticales en una variedad de especies que van desde aromáticas hasta ornamentales, pasando por especies con alto nivel nutricional. Una de las áreas de oportunidad de la empresa es el proceso de mantenimiento de los cultivos verticales, ya que estos deben de ser realizados in situ, por lo que se programan las visitas a las instalaciones de los clientes de acuerdo con las condiciones requeridas por las especies cultivadas, durante todo el año

Alcances y limitaciones.

Dentro de los alcances del prototipo se pueden definir que el sistema es capaz de realizar las lecturas de los sensores mediante Arduino para así poder transmitir los datos mediante la Shield Ethernet y poder alojar los datos en el servidor de MQTT.

El censado de las variables se realizan cada 5 minutos, para ellos se cuenta con la App para Android "*MQTT Dashborad"*, en la cual se pueden visualizar los datos en tiempo real, solo con suscribirse. Otra manera de visualizar los datos es acceder directamente al servidor MQTT, en el cual requiere identificación, es decir ingresar con un usuario y contraseña.

Las variables que este prototipo permite almacenar son:

- Humedad relativa.
- Temperatura relativa.
- Humedad del suelo.
- Iluminación.

En lo que respecta a la humedad del suelo permite la lectura en 3 niveles diferentes al igual que la iluminación se puede obtener los valores en 3 niveles.

Las limitaciones que tiene el prototipo, es que no cuenta con la instrucción de alarma en el caso de que un sensor no esté en línea es decir no esté en funcionamiento. Otra limitación es que el sistema no cuenta con un sistema de riego actualmente en el caso de que la humedad del suelo este debajo del mínimo definido, se tendrá que realizar el riego manual.

CAPITULO III.

Fundamento teórico.

Sistema microcontrolador.

Por todas las características que nos brinda sumado al bajo precio de mercado y a su versatilidad para realizar montajes electrónicos de control, registro y censado de variables físicas se optó por microcontrolador Arduino Mega 2560.

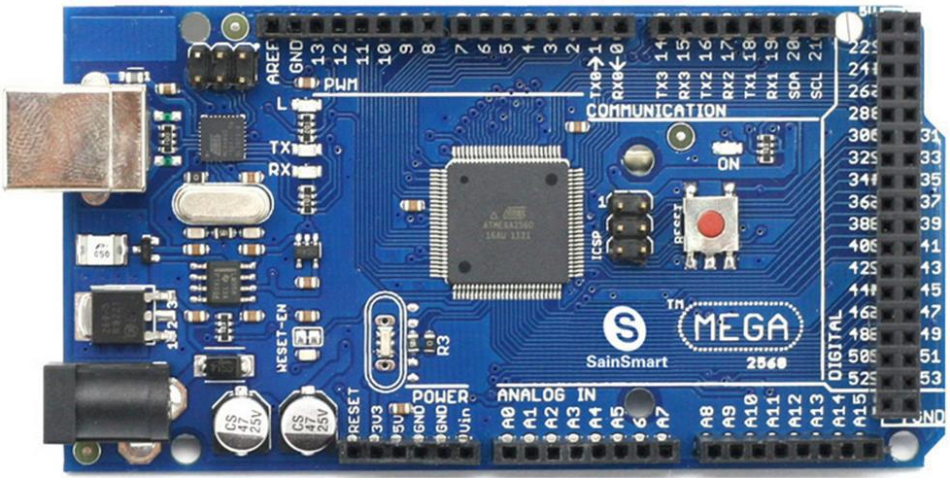


Imagen 1 Arduino Mega 2560.

Arduino es una plataforma de prototipos electrónicos de código abierto (open-source) basado en hardware y software flexibles y fáciles de usar. Está pensado para estudiantes, diseñadores, y para cualquiera interesado en crear objetos o entornos interactivos.

Arduino puede interpretar el entorno mediante la recepción de sus entradas desde una variedad de sensores y puede afectar su entorno mediante el control de transductores, luces, motores y otros artefactos. El Microcontrolador de la placa se programa usando el “Arduino Programming Language” (basado en Wiring1) y el “Arduino Development Environment” (basado en Processing2). Los proyectos de Arduino pueden ser autónomos o se pueden comunicar con software en ejecución en un ordenador, tableta o equipo móvil.

Las placas se pueden ensamblar a mano o encargarlas preensambladas; el software se puede descargar gratuitamente. Los diseños de referencia del

hardware están disponibles bajo licencia open-source, por lo que eres libre de adaptarlas a tus necesidades.

Arduino Mega posee las siguientes especificaciones:

Microcontrolador	ATmega2560
Tensión de funcionamiento	5V
Voltaje de entrada (recomendado)	7-12V
Voltaje de entrada (límite)	6-20V
Pernos digitales de E / S	54 (de los cuales 15 proporcionan salida PWM)
Clavijas de entrada analógica	Dieciséis
Corriente DC por Pin de E / S	20 mA
Corriente DC para 3.3V Pin	50 mA
Memoria flash	256 KB de los cuales 8 KB utilizados por el gestor de arranque
SRAM	8 KB
EEPROM	4 KB
Velocidad de reloj	16 MHz
LED_BUILTIN	13
Longitud	101.52 mm
Anchura	53.3 mm
Peso	37 g

Alimentación.

Arduino Mega puede ser alimentado mediante el puerto USB o con una fuente externa de poder. La alimentación es seleccionada de manera automática.

Cuando se trabaja con una fuente externa de poder se debe utilizar un convertidor AC/DC y regular dicho voltaje en el rango operativo de la placa. De igual manera se puede alimentar el micro mediante el uso de baterías. Preferiblemente el voltaje debe estar en el rango de los 7V hasta los 12V.

Arduino Mega posee algunos pines para la alimentación del circuito aparte del adaptador para la alimentación:

- VIN: A través de este pin es posible proporcionar alimentación a la placa.
- 5V: Podemos obtener un voltaje de 5V y una corriente de 40mA desde este pin.
- 3.3V: Podemos ubtener un voltaje de 3.3V y una corriente de 50mA desde este pin.
- GND: El ground (0V) de la placa.

Arduino puede ser programado de una manera muy fácil utilizando el lenguaje propio de Arduino junto con la interfaz Arduino IDE.

Programación.

La placa Mega 2560 se puede programar con el software Arduino (IDE).

El ATmega2560 en el Mega 2560 viene pre programado con un cargador de arranque que le permite cargar un nuevo código sin el uso de un programador de hardware externo. Se comunica utilizando el protocolo STK500 original (referencia, archivos de cabecera C). También puede omitir el gestor de arranque y programar el microcontrolador a través del encabezado ICSP (Programación en Serie en Circuito) utilizando el ISP de Arduino o similar. El código fuente del firmware ATmega16U2 (o 8U2 en las placas rev1 y rev2) está disponible en el repositorio Arduino. El ATmega16U2 / 8U2 se carga con un gestor de arranque DFU, que puede activarse mediante:

- En las placas Rev1: conecte el puente de soldadura en la parte posterior de la placa (cerca del mapa de Italia) y luego reinicie el 8U2.
- En las placas Rev2 o posteriores: hay una resistencia que tira de la línea HWB 8U2 / 16U2 a tierra, lo que facilita la puesta en modo DFU. Luego puede usar el software FLIP de Atmel (Windows) o el programador DFU (Mac OS X y Linux) para cargar un nuevo firmware. O puede usar el encabezado ISP con un programador externo (sobrescribiendo el cargador de arranque DFU).

Ethernet Shield.

El Arduino Ethernet Shield permite a una placa Arduino conectarse a internet. Está basada en el chip ethernet Wiznet W5100. El Wiznet W5100 provee de una pila de red IP capaz de TCP y UDP. Soporta hasta cuatro conexiones de sockets simultáneas. Usa la librería Ethernet para escribir programas que se conecten a internet usando la shield.

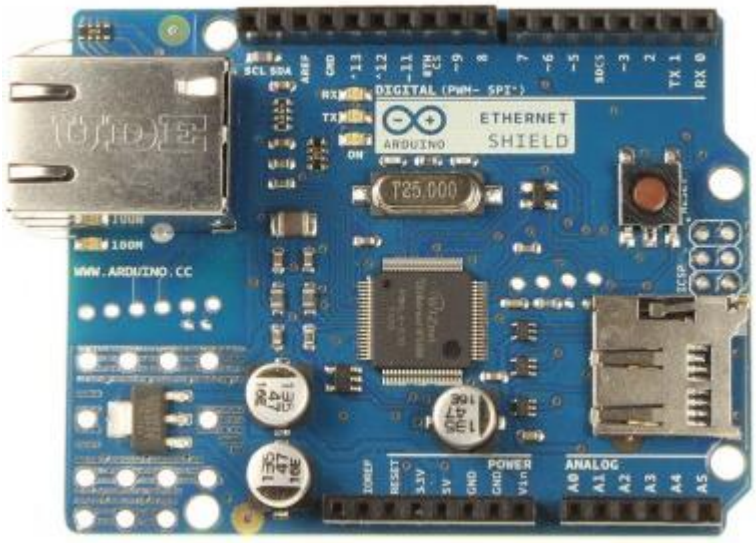


Imagen 2 Arduino Ethernet Shield R3.

Es compatible con el Arduino UNO y Arduino Mega.

El shield provee un conectore ethernet estándar RJ45 y un conector lector de tarjeta Micro SD

El botón de reset en la shield resetea ambos, el W5100 y la placa Arduino.

- El shield contiene un número de LEDs para información:
- PWR: indica que la placa y la shield están alimentadas
- LINK: indica la presencia de un enlace de red y parpadea cuando la shield envía o recibe datos
- FULLD: indica que la conexión de red es full duplex
- 100M: indica la presencia de una conexión de red de 100 Mb/s (de forma opuesta a una de 10Mb/s)
- RX: parpadea cuando la shield recibe datos
- TX: parpadea cuando la shield envía datos
- COLL: parpadea cuando se detectan colisiones en la red.

El Arduino Ethernet Shield 2 conecta su Arduino a Internet en cuestión de minutos. Simplemente conecte este módulo a su placa Arduino, conéctelo a su red con un cable RJ45 (no incluido) y siga unos simples pasos para comenzar a controlar su mundo a través de Internet. Como siempre con Arduino, cada elemento de la plataforma (hardware, software y documentación) está disponible de forma gratuita y de código abierto. Esto significa que puede aprender exactamente cómo se hace y usar su diseño como punto de partida para sus propios circuitos. Cientos de miles de tableros Arduino ya están alimentando la creatividad de las personas en todo el mundo, todos los días.

- Requiere un tablero Arduino (no incluido)
- Voltaje de funcionamiento 5V (suministrado desde la placa Arduino)
- Controlador Ethernet: W5500 con búfer interno 32K
- Velocidad de conexión: 10 / 100Mb.
- Conexión con Arduino en el puerto SPI

El Arduino Ethernet Shield 2 permite que una placa Arduino se conecte a Internet. Se basa en el (chip Wiznet W5500 Ethernet). El Wiznet W5500 proporciona una pila de red (IP) capaz de TCP y UDP. Admite hasta ocho conexiones de socket simultáneas. Use la biblioteca de Ethernet para escribir bocetos que se conectan a Internet usando el escudo. El Ethernet Shield 2 se conecta a una placa Arduino utilizando cabezales de envoltura de alambre largo que se extienden a través del Shield. Esto mantiene intacto el diseño del pin y permite que otro escudo se apile sobre él. La revisión más reciente de la placa expone el pinout 1.0 en la versión 3 de la Junta Arduino UNO. El Ethernet Shield 2 tiene una conexión RJ-45 estándar, con un transformador de línea integrado y Power over Ethernet habilitado. Hay una ranura para tarjeta micro-SD integrada, que se puede utilizar para almacenar archivos para servir a través de la red. Es compatible con Arduino Uno y Mega (utilizando la biblioteca de Ethernet). Se puede acceder al lector de tarjetas micro-SD a bordo a través de la biblioteca SD. Cuando se trabaja con esta biblioteca, SS está en el Pin 4. La revisión original del Shield contenía una ranura para tarjeta SD de tamaño completo; Esto no es compatible. El protector también incluye un controlador de reinicio, para garantizar que el módulo Ethernet W5500 se reinicie correctamente en el encendido. Las revisiones anteriores del Shield no eran compatibles con el Mega y debían reiniciarse manualmente después del encendido. El escudo actual tiene un módulo de alimentación a través de

Ethernet (PoE) diseñado para extraer energía de un cable Ethernet de categoría 5 de par trenzado convencional.

Las características del módulo PoE son las siguientes:

- Cumple con IEEE802.3af
- Rango de voltaje de entrada 36V a 57V
- Protección contra sobrecargas y cortocircuitos.
- Salida 12V
- Convertidor CC / CC de alta eficiencia: tipo 85% a 80% de carga
- Aislamiento 1500V (entrada a salida)

El Shield no viene con un módulo PoE incorporado, es un componente separado que debe agregarse. Arduino se comunica tanto con la tarjeta W5500 como con la tarjeta SD utilizando el bus SPI (a través del encabezado ICSP). Esto está en los pines digitales 10, 11, 12 y 13 en el Uno y los pines 50, 51 y 52 en el Mega. En ambas tarjetas, el pin 10 se utiliza para seleccionar el W5500 y el pin 4 para la tarjeta SD. Estos pines no se pueden utilizar para E / S general. En el Mega, el pin SS del hardware, 53, no se usa para seleccionar el W5500 o la tarjeta SD, pero debe guardarse como salida o la interfaz SPI no funcionará. Tenga en cuenta que dado que la tarjeta W5500 y la tarjeta SD comparten el bus SPI, solo puede estar activo uno a la vez. Si está utilizando ambos periféricos en su programa, esto debe ser cuidado por las bibliotecas correspondientes. Sin embargo, si no está utilizando uno de los periféricos en su programa, deberá deseleccionarlo explícitamente. Para hacer esto con la tarjeta SD, establezca el pin 4 como salida y escriba un alto para ello. Para el W5500, configure el pin digital 10 como una salida alta.

- El Shield proporciona un conector RJ45 Ethernet estándar.
- El botón de reinicio en el Escudo reinicia tanto el W5500 como la placa Arduino.
- El escudo contiene una serie de indicadores LED de información:
- ENCENDIDO: indica que la placa y el escudo están alimentados
- ENLACE: indica la presencia de un enlace de red y parpadea cuando el escudo transmite o recibe datos
- FDX: indica que la conexión de red es full duplex
- 100M: indica la presencia de una conexión de red de 100 Mb / s (en lugar de 10 Mb / s)
- ACT: parpadea cuando hay actividad RX o TX

Higrómetro fc-28.

Un higrómetro de suelo FC-28 es un sensor que mide la humedad del suelo. Son ampliamente empleados en sistemas automáticos de riego para detectar cuando es necesario activar el sistema de bombeo.

¿Cómo está compuesto el paquete?

- 1 x módulo sensor de humedad del suelo FC-28.
- 1 x placa de control.
- 2 x cables macho – hembra para la conexión del sensor.

El FC-28 es un sensor sencillo que mide la humedad del suelo por la variación de su conductividad. No tiene la precisión suficiente para realizar una medición absoluta de la humedad del suelo, pero tampoco es necesario para controlar un sistema de riego.

El FC-28 se distribuye con una placa de medición estándar que permite obtener la medición como valor analógico o como una salida digital, activada cuando la humedad supera un cierto umbral.

Los valores obtenidos van desde 0 sumergido en agua, a 1023 en el aire (o en un suelo muy seco). Un suelo ligeramente húmero daría valores típicos de 600-700. Un suelo seco tendrá valores de 800-1023.

Imagen 3 HIGRÓMETRO FC-28.

Esquema de montaje.

El esquema eléctrico es sencillo. Alimentamos el módulo conectando GND y 5V a los pines correspondientes de Arduino.

Ahora si queremos usar la lectura analógica, conectamos la salida A0 a una de las entradas analógicas de Arduino.

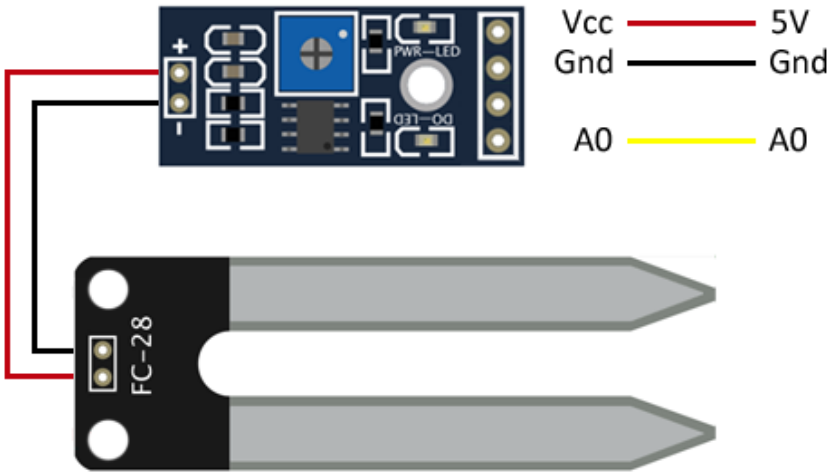


Imagen 4 ESQUEMA DE MONTAJE.

Si quisiéramos emplear el valor digital, que se ajusta con el potenciómetro de la placa, en su lugar conectaríamos la salida D0 del sensor a una entrada digital de Arduino.

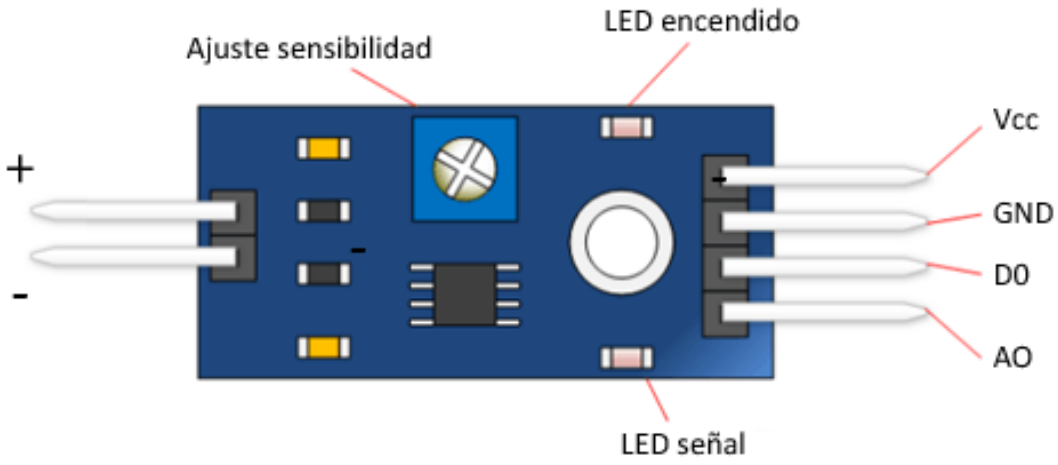


Imagen 5 Descripción de Higrómetro.

Modulo sensor de intensidad de luz LDR.

El funcionamiento de un fotoresistor se basa en el efecto fotoeléctrico. Un fotoresistor está hecho de un semiconductor de alta resistencia como el sulfuro de cadmio, CdS. Si la luz que incide en el dispositivo es de alta frecuencia, los fotones son absorbidos por las elasticidades del semiconductor dando a los electrones la suficiente energía para saltar la banda de conducción. El electrón libre que resulta, y su hueco asociado, conducen la electricidad, de tal modo que disminuye la resistencia. Los valores típicos varían entre 1 MOhm, o más, en la oscuridad y 100 Ohm con luz brillante.

La variación del valor de la resistencia tiene cierto retardo, diferente si se pasa de oscuro a iluminado o de iluminado a oscuro. Esto limita a no usar los LDR en aplicaciones en las que la señal luminosa varia con rapidez. El tiempo de respuesta típico de un LDR está en el orden de una décima de segundo. Esta lentitud da ventaja en algunas aplicaciones, ya que se filtran variaciones rápidas de iluminación que podrían hacer inestable un sensor (ej. tubo fluorescente alimentado por corriente alterna). En otras aplicaciones (saber si es de día o es de noche) la lentitud de la detección no es importante.

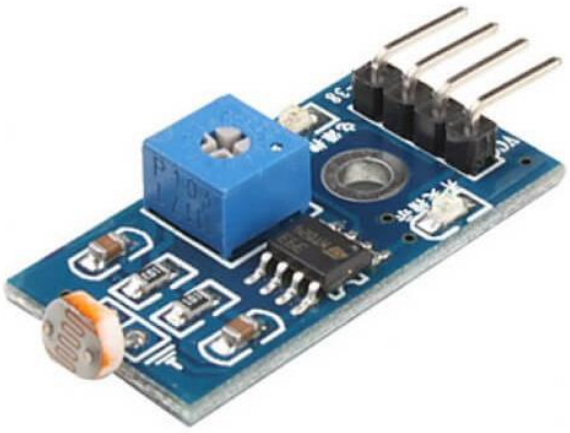


Imagen 6 sensor de presión BMP180.

Especificaciones técnicas.

- Voltaje de Operación: 5V DC
- Conexión de 4 cables: VCC, GND, DO, AO
- Salida analógica y digital(comparador)
- Opamp en modo comparador: LM393
- Potenciómetro para ajuste de comparador
- Led rojo de encendido y verde de salida digital.

Sensor de temperatura y humedad DHT22 o AM2302.

El sensor de temperatura y humedad DHT22 o AM2302 se une a la familia de los sensores DHT, de los que hasta ahora solo conocíamos a su hermano pequeño el sensor de temperatura y humedad DHT11. El sensor DHT22 supone una mejora considerable en las características técnicas con respecto al DHT11 a cambio de un pequeño incremento en el precio.

Encapsulados:

- El sensor suelto, con una “funda” blanca plástica y cuatro pines de conexión.
- El sensor con la misma funda blanca que el anterior, pero esta vez soldado en una placa y con tres pines de conexión, además de una resistencia pull-up (entre 3-6 kΩ) y un condensador de filtrado (normalmente de 100 nF).

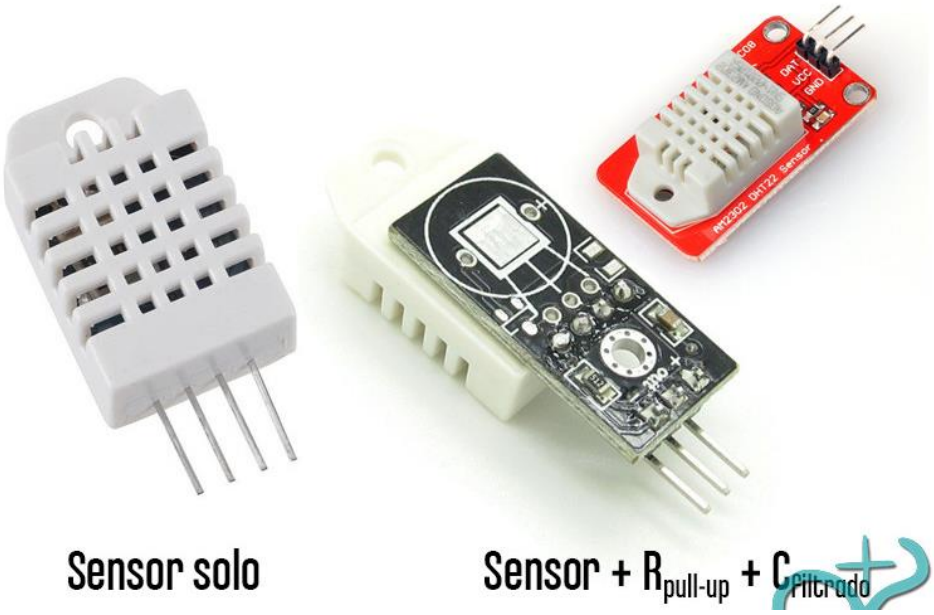


Imagen 7 Conexiones entre Arduino y módulo BMP180.

Características:

- Potencia ultra baja
- Calibración totalmente automatizada
- Sensor de humedad capacitivo
- Salida: standard digital single-bus

- Voltaje de funcionamiento: 3.3 V ~ 5.5 V
- Corriente en medición: 500 uA típicamente
- Alta precisión: ± 0.5 °C típicamente para la temperatura y ± 2 %RH para humedad
- Resolución: Temperatura: 0.1 °C típicamente; Humedad: 0.1 %RH
- Rangos de operación: Temperatura: -40°C ~ 80°C; Humedad: 0 ~ 99.9 %RH

Esquema montaje.

La conexión del DHT11 y el DHT22 son idénticas, ya que como hemos comentado la única diferencia entre modelos son sus prestaciones. En ambos casos, disponemos de 4 patillas, de las cuales usaremos 3, Vcc, Output y GND.

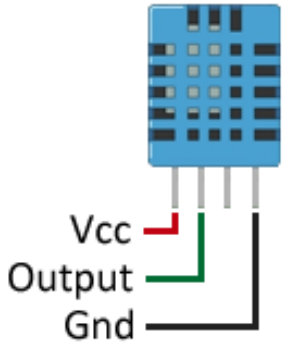


Imagen 8 Pines de DHT 11 / DHT 22.

Conectar el sensor es sencillo, simplemente alimentamos desde Arduino al sensor a través de los pines GND y Vcc del mismo. Por otro lado, conectamos la salida Output a una entrada digital de Arduino. Necesitaremos poner una resistencia de 10K entre Vcc y el Pin Output.

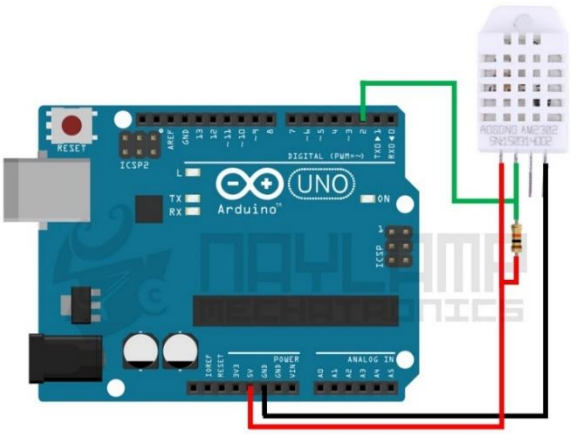


Imagen 9 Conexiones entre Arduino y el sensor DHT22

La resistencia Pull-Up puede ser un valor entre 4.7K y 10K. Si se desea trabajar con lógica de 3.3v solo hay que cambiar la alimentación a dicho voltaje al igual que la resistencia pull-up debe ir a 3.3V, en nuestro caso vamos a trabajar con el pin digital 2, pero pueden usar otro pin si lo desean.

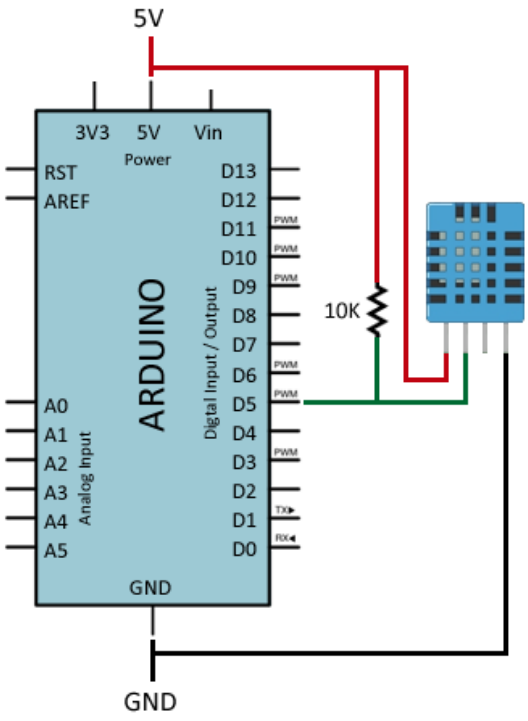


Imagen 10 El esquema eléctrico queda como la siguiente imagen.

Protocolo MQTT: (Message Queue Telemetry Transport).
Es un protocolo usado para la comunicación machine-to-machine (M2M) en el "Internet of Things". Este protocolo está orientado a la comunicación de sensores, debido a que consume muy poco ancho de banda y puede ser utilizado en la mayoría de los dispositivos empotrados con pocos recursos (CPU, RAM, …). Un ejemplo de uso de este protocolo es la aplicación de Facebook Messenger tanto para android y Iphone. La arquitectura de MQTT sigue una topología de estrella, con un nodo central que hace de servidor o "broker" con una capacidad de hasta 10000 clientes. El broker es el encargado de gestionar la red y de transmitir los mensajes, para mantener activo el canal, los clientes mandan periódicamente un paquete (PINGREQ) y esperan la respuesta del broker (PINGRESP). La comunicación puede ser cifrada entre otras muchas opciones.

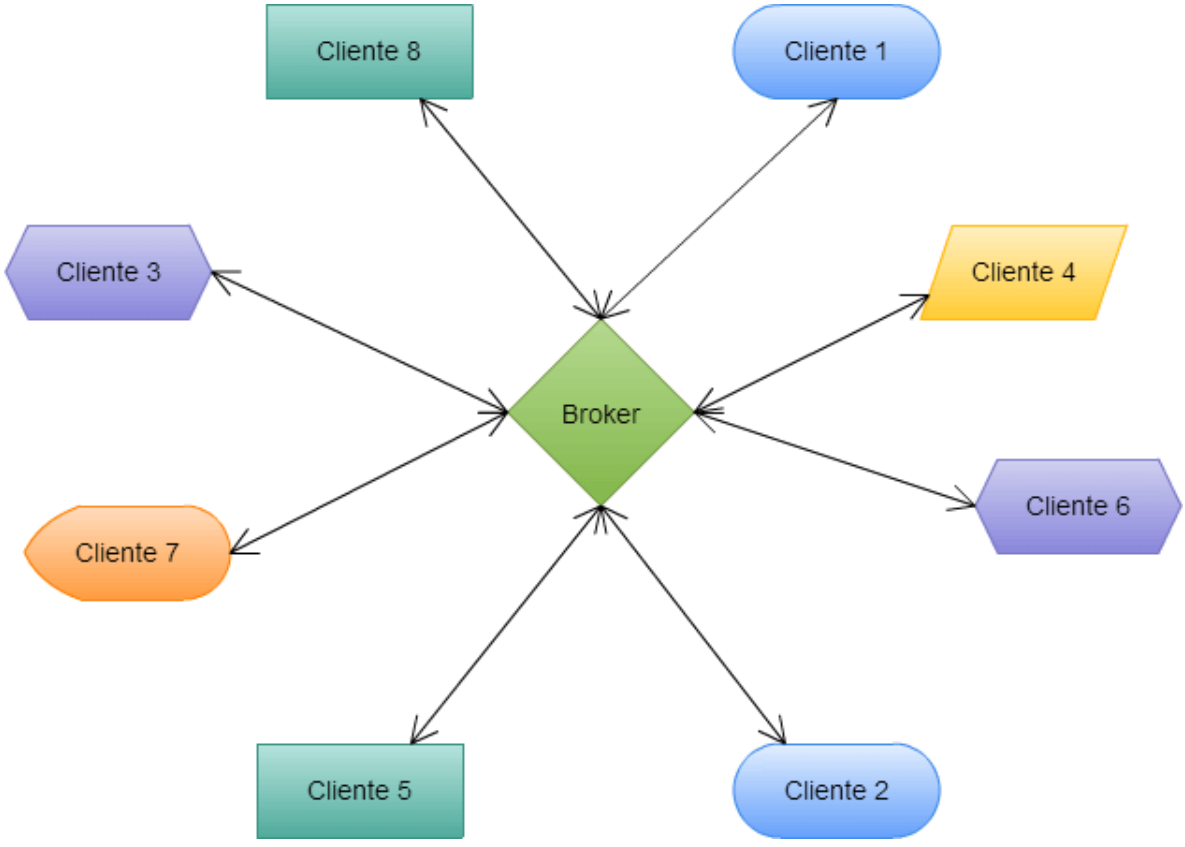


Imagen 11 Topología MQTT.

La comunicación se basa en unos "topics" (temas), que el cliente que publica el mensaje crea y los nodos que deseen recibirlo deben subscribirse a él. La comunicación puede ser de uno a uno, o de uno a muchos. Un "topic" se representa mediante una cadena y tiene una estructura jerárquica. Cada jerarquía se separa con '/'.

Por ejemplo, "edificio1/planta5/sala1/raspberry2/temperatura" o "/edificio3/planta0/sala3/arduino4/ruido".

De esta forma se pueden crear jerarquías de clientes que publican y reciben datos, como podemos ver en la imagen:

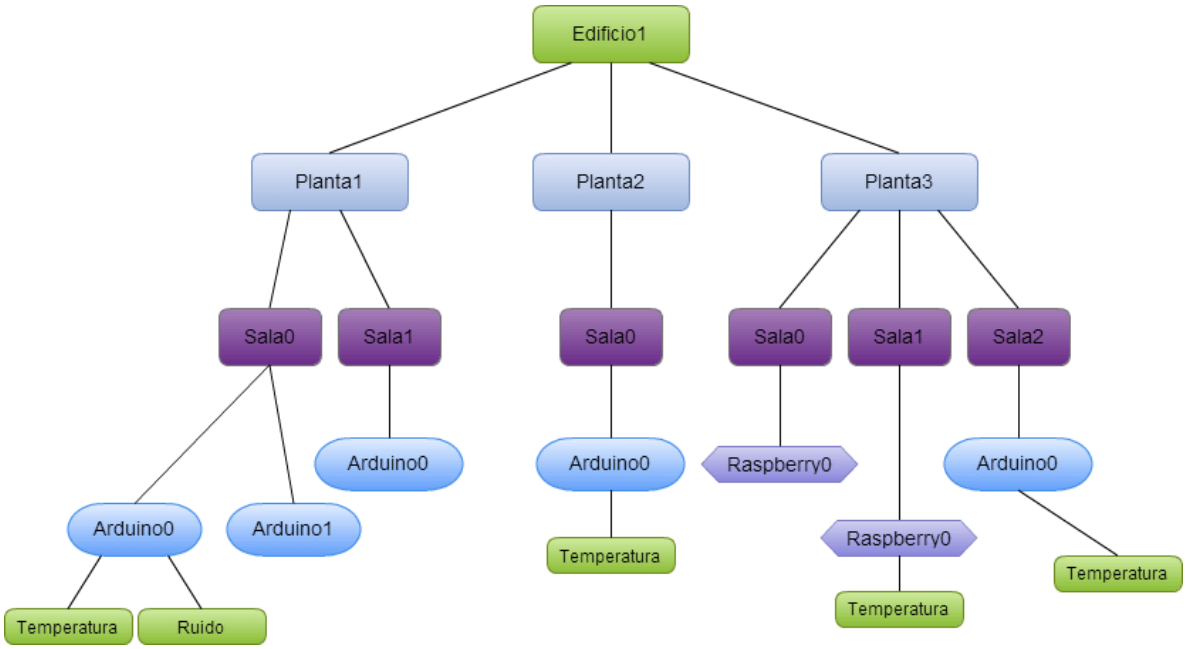


Imagen 12 Jerarquía MQTT.

De esta forma un nodo puede subscribirse a un "topic" concreto ("edificio1/planta2/sala0/arduino0/temperatura") o a varios ("edificio1/planta2/#").

¿Por qué MQTT y no otro?

MQTT es un protocolo abierto, sencillo, ligero y fácil de implantar.

Es ideal para responder a las siguientes necesidades:

- Está especialmente adaptado para utilizar un ancho de banda mínimo
- Es ideal para utilizar redes inalámbricas
- Consume muy poca energía
- Es muy rápido y posibilita un tiempo de respuesta superior al resto de protocolos web actuales
- Permite una gran fiabilidad si es necesario
- Requiere pocos recursos procesadores y memorias
- El MQTT no es el único protocolo que intenta imponerse: otros como XMPP, REST API y CoAp también tienen ciertas ventajas.

Seguridad.

Los datos de IoT intercambiados pueden resultar muy críticos, por lo que es posible garantizar la seguridad de los intercambios en varios niveles:

- Transporte en SSL/TLS
- Autenticación mediante certificados SSL/TLS
- Autenticación mediante usuario y contraseña.

El modelo de publicación y suscripción.

El protocolo MQTT define los tipos de entidades en la red: un intermediario de mensajes y un número de clientes. El intermediario es un servidor que recibe todos los mensajes de los clientes y luego los redirige a clientes de destinos relevantes. Un cliente es cualquier cosa que pueda interactuar con el intermediario para enviar y recibir mensajes. Un cliente puede ser un sensor de IoT en el campo o una aplicación del centro de datos que procesa datos de IoT.

1. El cliente se conecta con el intermediario. Se puede suscribir a cualquier "tema" de mensajes de intermediario. Esta conexión puede ser una conexión TCP/IP simple o una conexión TLS cifrada para mensajes confidenciales.
2. El cliente publica el mensaje, sobre un tema, enviando el mensaje y el tema al intermediario.

3. Después, el intermediario redirige el mensaje a todos los clientes que están suscritos a ese tema.

Ya que los mensajes MQTT están organizados por temas, el desarrollador de aplicaciones tiene la flexibilidad de especificar que ciertos clientes sólo puedan interactuar con determinados mensajes. Por ejemplo, los sensores publicarán sus lecturas sobre del tema "sensor_data" y se suscribirán al tema "config_change". Las aplicaciones de procesamiento de datos que guardan datos del sensor en una base de datos backend se suscribirán al tema "sensor_data". Una aplicación de consola de administración puede recibir los comandos del administrador del sistema para ajustar las configuraciones de los sensores, como la sensibilidad y la frecuencia de la muestra, y publicar esos cambios en el tema "config_change"

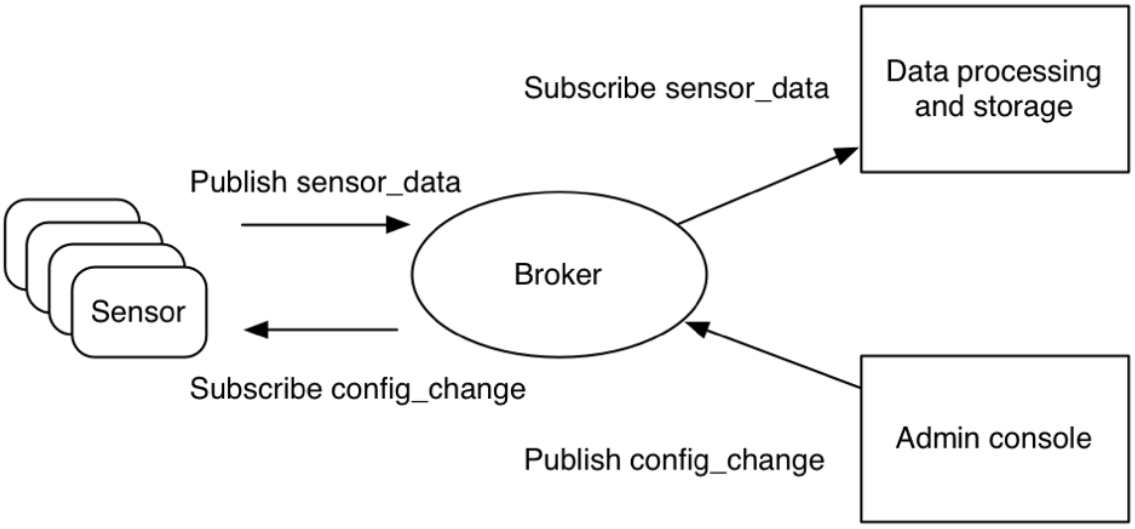


Imagen 13 modelo de publicación y suscripción MQTT para sensores IoT.

Desarrollo.

Programación de los sensores en Arduino.

Como primer paso ingresamos a www.arduinoblocks.com (Imagen 14)

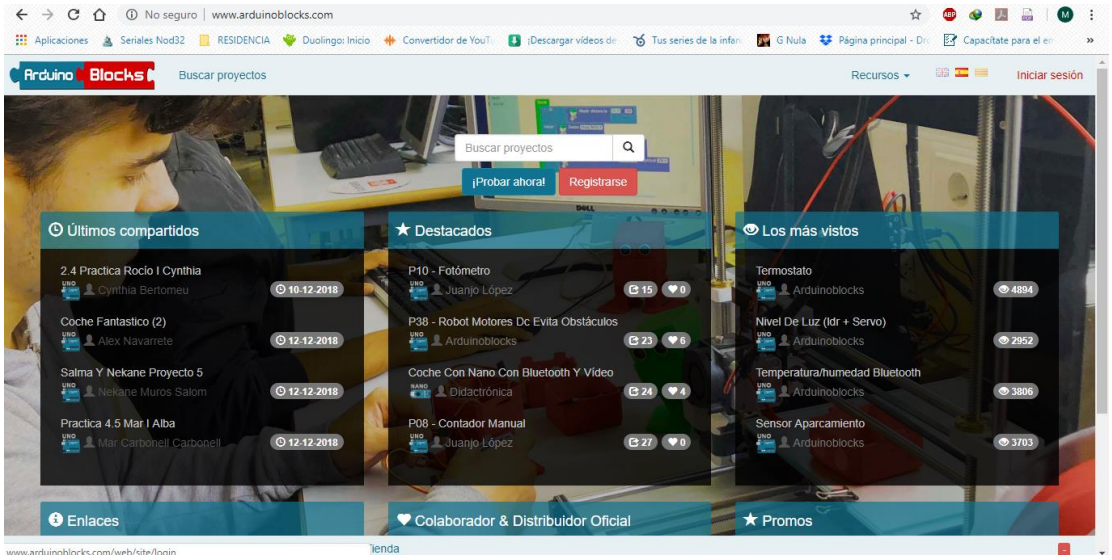


Imagen 14 ingreso a la página web.

A continuación, es necesario ingresar con una cuenta valida. (Imagen 15).

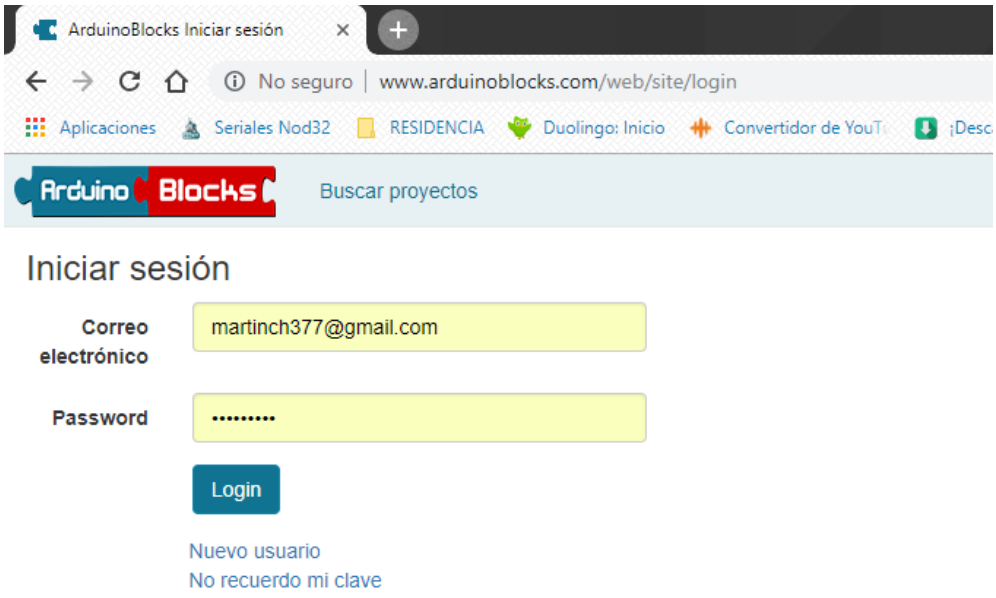


Imagen 15 Inicio de sesión.

Ya iniciado la sesión, se creará un nuevo proyecto.

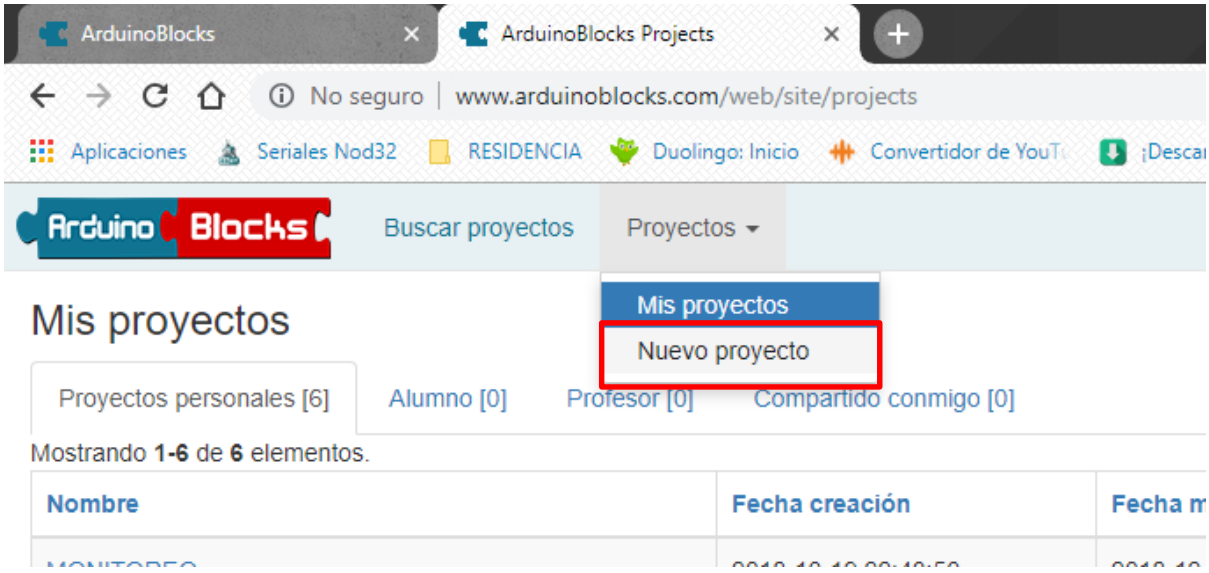


Imagen 16 Creación de un nuevo proyecto.

En la siguiente ventana nos presenta el tipo de proyecto que se puede crear. En este caso será un proyecto personal. Por lo que seleccionaremos *iniciar proyecto personal*. (Imagen 17).

Imagen 17 Selección del tipo de proyecto a crear.

En esta sección hay que definir el tipo de placa con el cual se trabajará, en este caso será con Arduino Mega / 2560 (Imagen 18).

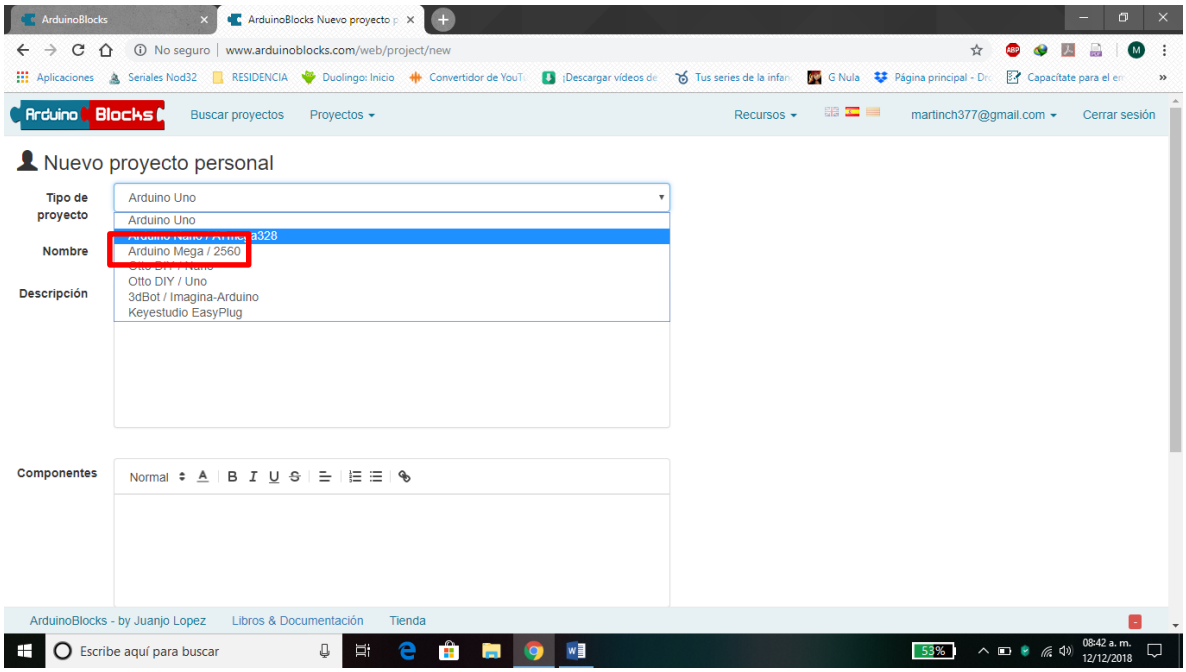


Imagen 18 Elección del tipo de placa con la que se trabajara.

También se definirá el nombre que llevará el proyecto en nuestro caso será nombrado como "*MONITOREO*" (Imagen 19).

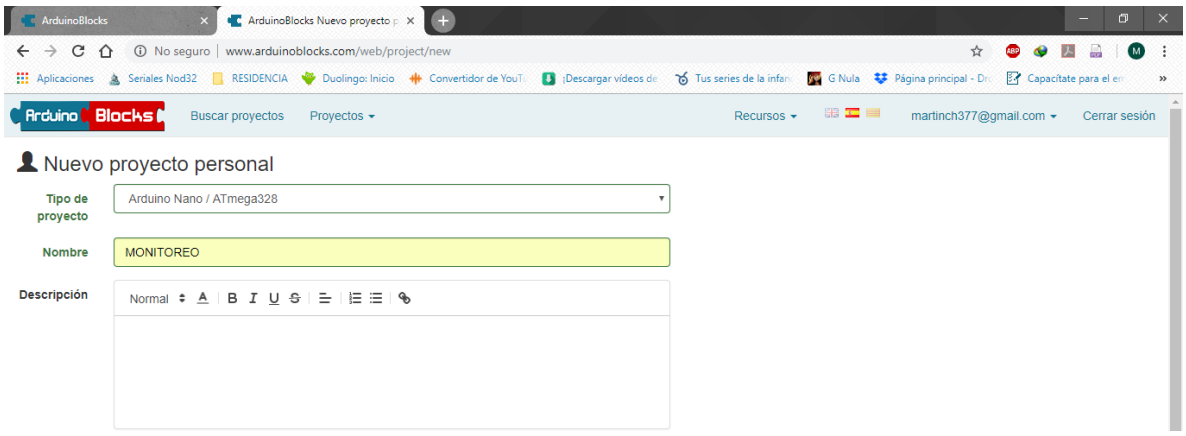


Imagen 19 Definición del nombre del proyecto.

Ventana inicial del proyecto, aquí se puede observar que tenemos dos componentes “*Inicializar*” y “*Bucle*”. (Imagen 20).

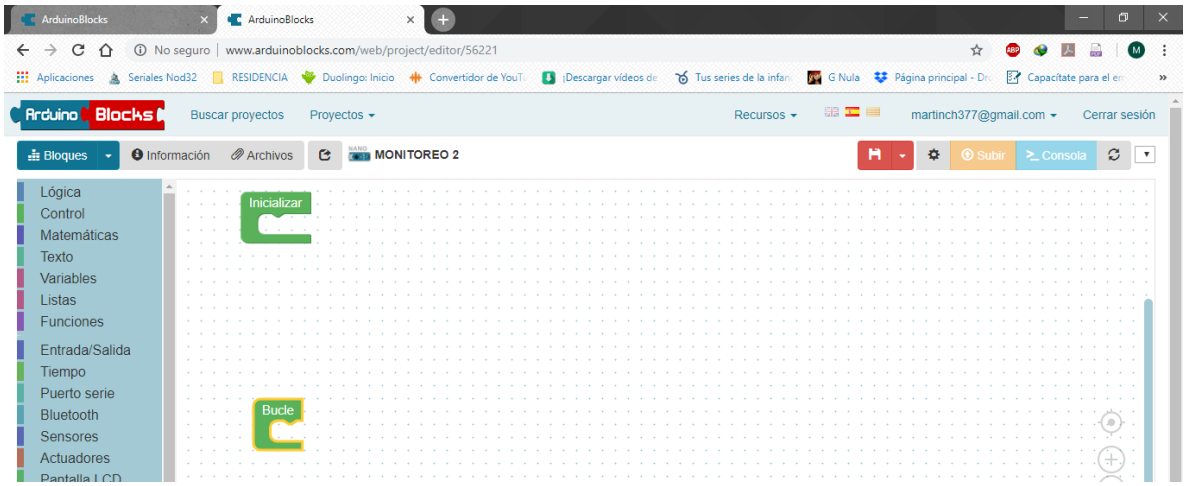


Imagen 20 Ventana principal del proyecto.

Para iniciar con lo que sería el programa, primero nos dirigimos a la sección *MQTT*, para elegir la opción “*Iniciar (Ethernet Shield)*”, para así poder conectarnos a un servidor mediante ethernet (Imagen 21).

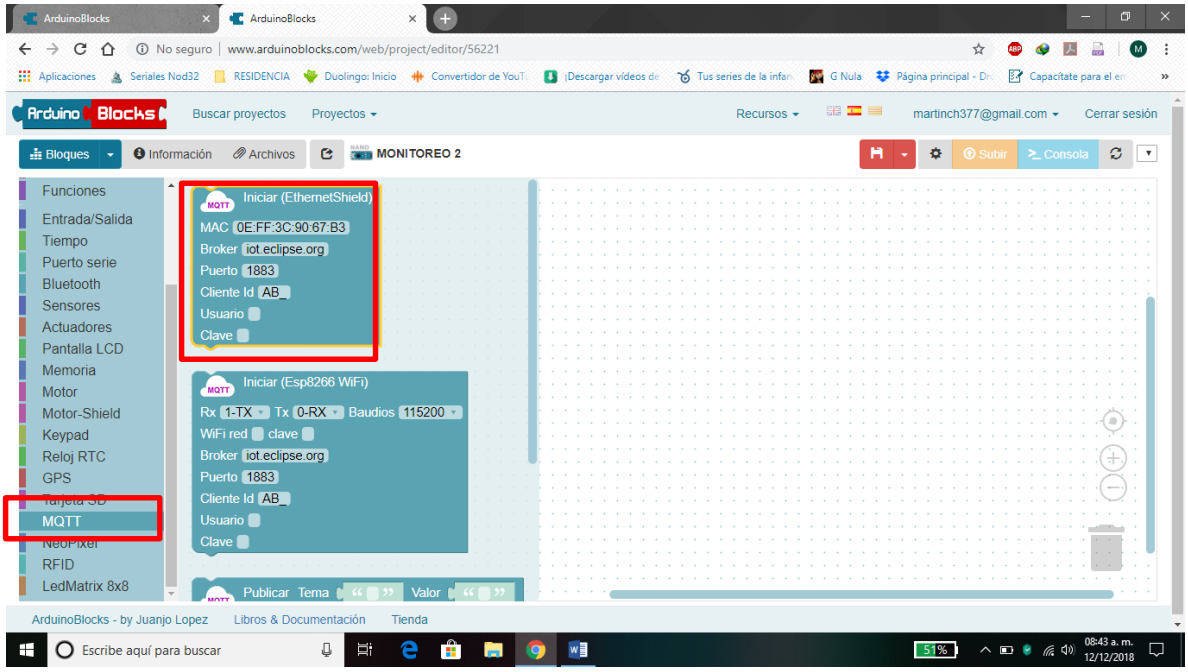


Imagen 21 Sección MQTT.

Ya que elegimos el bloque de *“Iniciar (Ethernet)”,* lo colocaremos en la sección de *“Inicializar*” para así cuando el programa corra se pueda conectar al servidor que alojara los datos (Imagen 22).

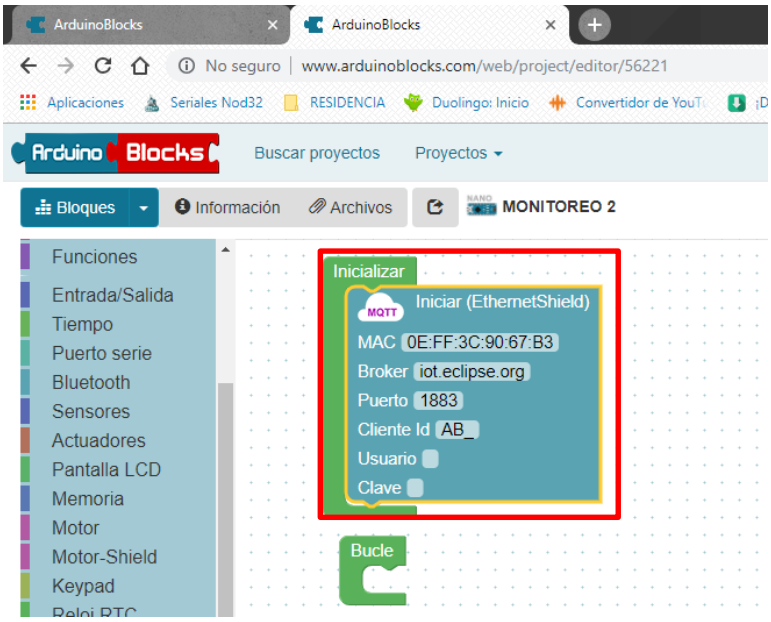


Imagen 22 Inicializar el programa.

Para poder hacer que el programa realice la lectura de los sensores cada determinado tiempo, vamos a elegir el bloque “*ejecutar cada*” el cual lo encontramos en la sección de tiempo (Imagen 23).

Imagen 23 Creación de un ciclo.

El bloque “*Ejecutar cada*” se ubicará en la sección “*Bucle*” para que el programa sea cíclico y así podamos tener en determinado tiempo las lecturas de los sensores (Imagen 24).

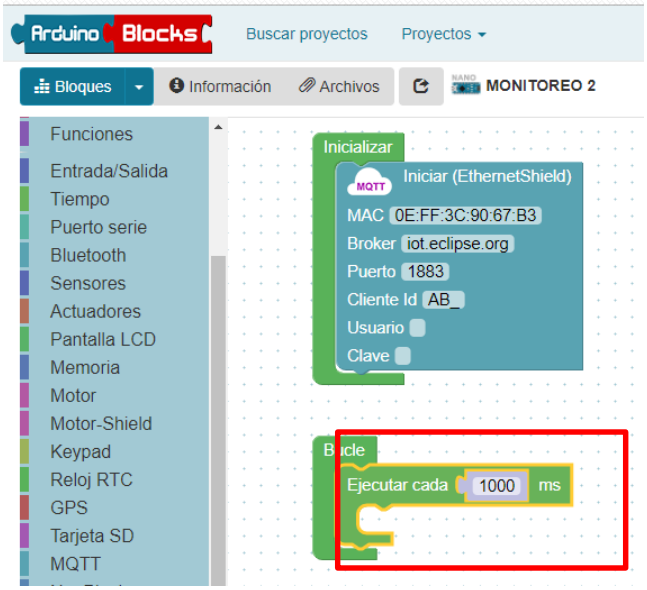


Imagen 24 Determinación de tiempo del ciclo.

Ahora para poder enviar los datos obtenidos de las lecturas de los sensores al servidor, se utilizará el bloque “*publicar tema*” (Imagen 25).

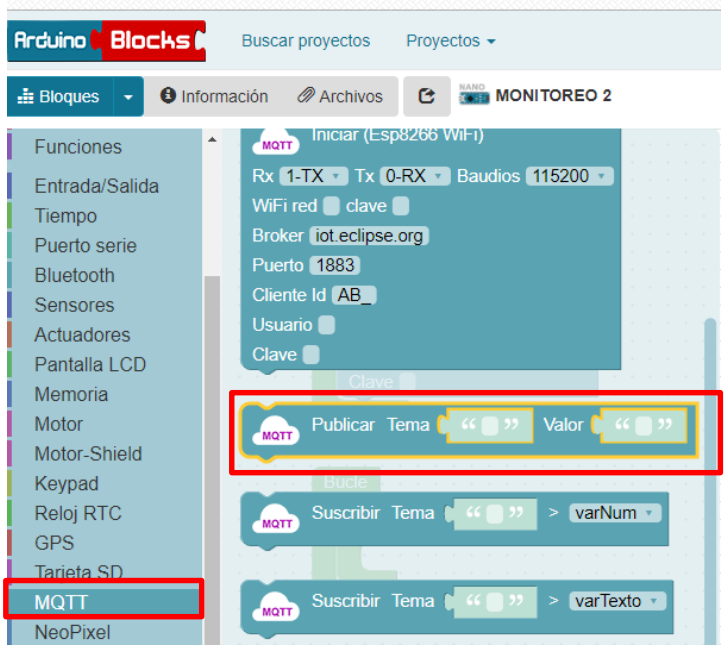


Imagen 25 Bloque para enviar datos al servidor.

El bloque “*Publicar Tema*” será ubicado en el bloque “*Ejecutar cada*” para que poder tener la lectura de manera cíclica (Imagen 26).

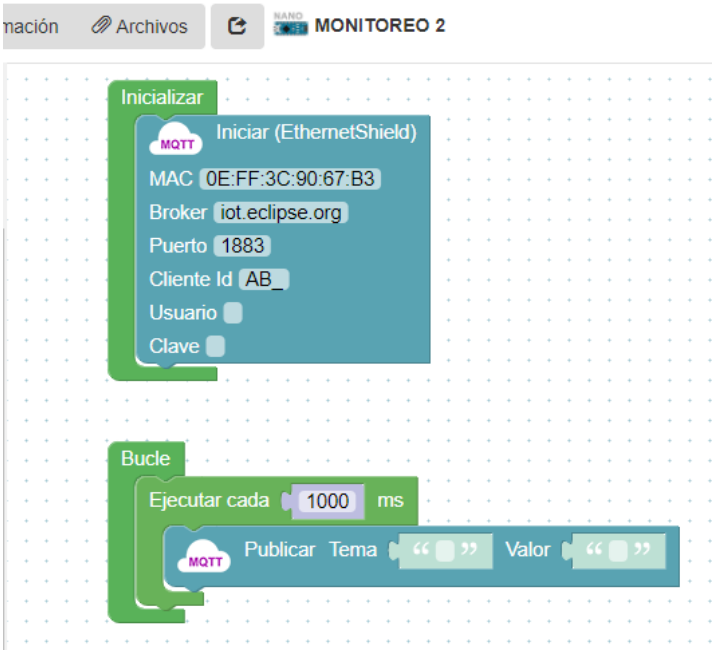


Imagen 26 Ingreso al ciclo.

Ahora se agregarán uno de los sensores a usar, para ello nos dirigimos a la sección de sensores para ubicar al sensor DHT-22 el cual nos proporcionara la temperatura y humedad (Imagen 27).

Imagen 27 Sensor DHT-22.

La que localizamos el sensor DHT-22, lo ubicaremos en el bloque “*Ejecutar cada*” para poder realizar la instrucción de leer el sensor DHT-22 en el pin 2, y ahí debemos definir el nombre que llevara esa instrucción “AB/temperatura” (Imagen 28).

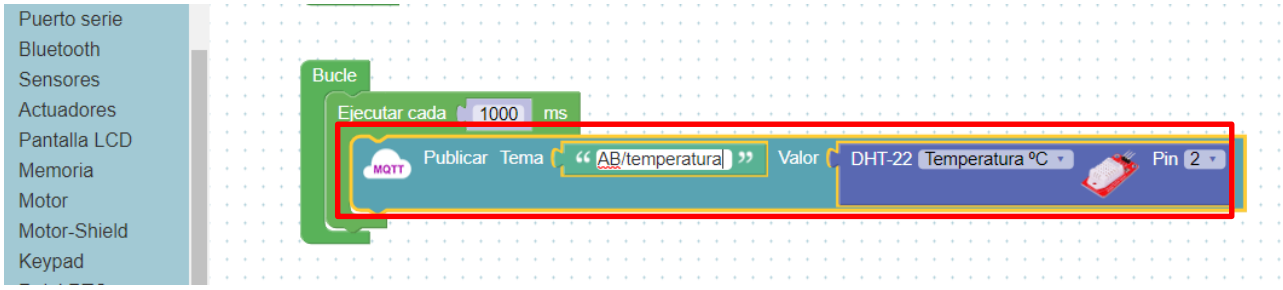


Imagen 28 Instrucción para leer la temperatura.

Ahora necesitamos realizar la lectura de la humedad relativa que nos proporciona el DHT-22, ahora que ya tenemos la lectura de la temperatura, duplicaremos esa instrucción (Imagen 29).

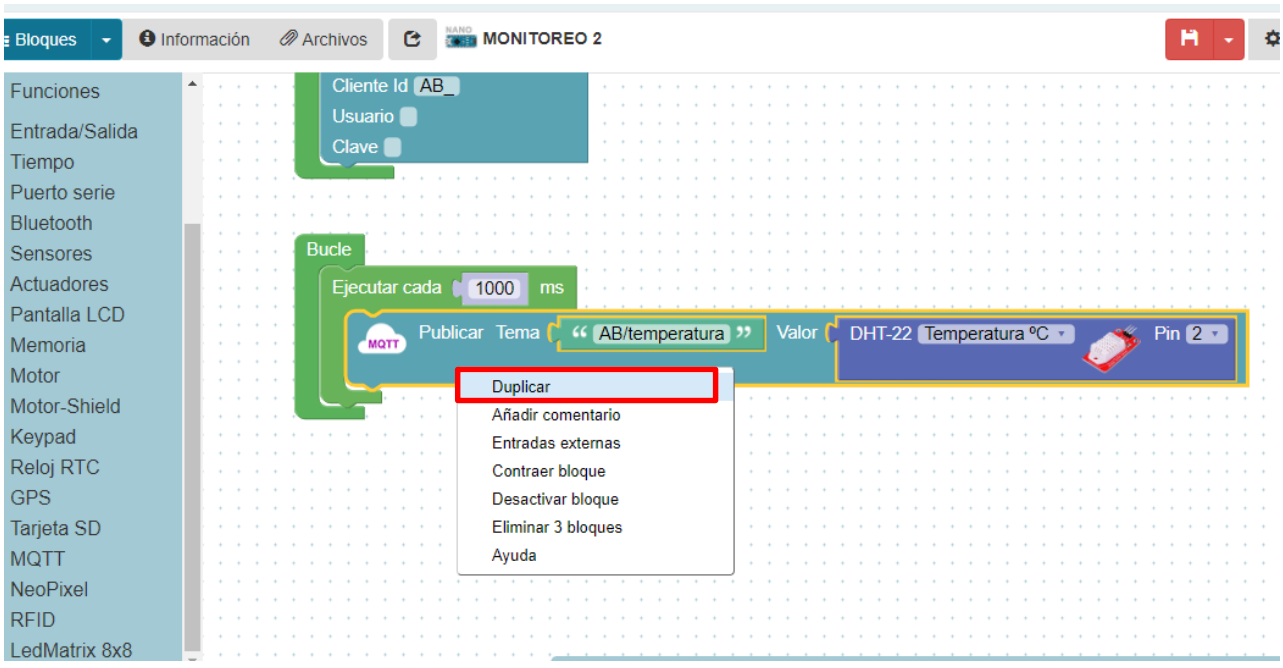


Imagen 29 Duplicar instrucción.

Ya que tenemos la instrucción duplicada cambiaremos el parámetro de lectura del DHT-22 el cual será la humedad, de igual manera el nombre de la instrucción cambiará a *"AB/humedad*" para así poder leer también la humedad relativa (Imagen 30).

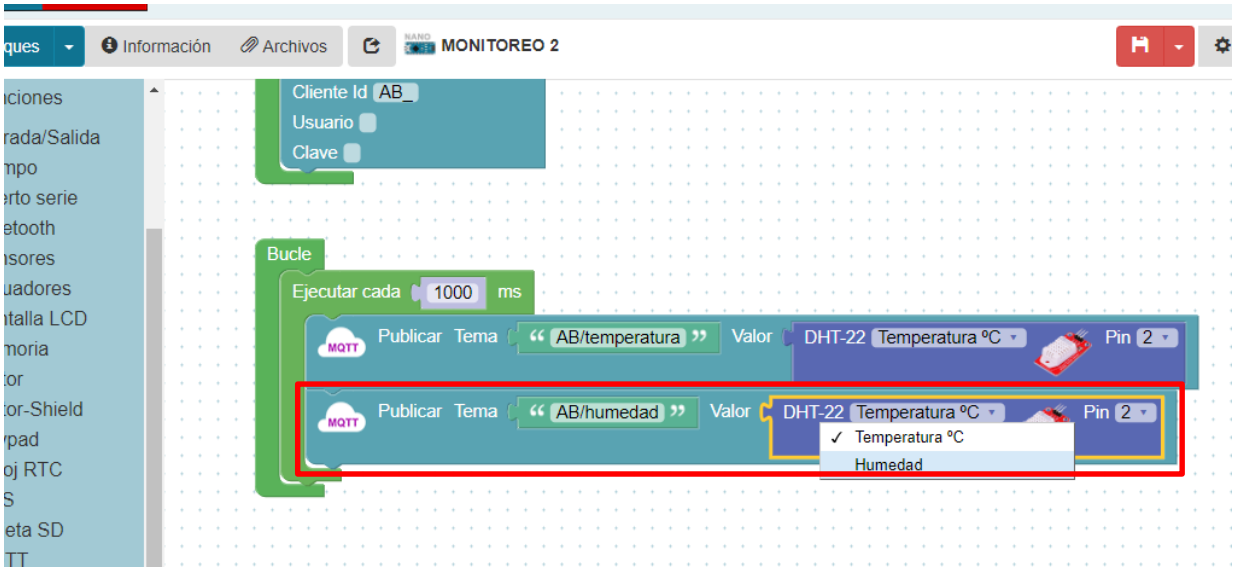


Imagen 30 Lectura de la humedad relativa.

Buscamos el sensor de humedad, en la sección de sensores, así también se utilizará el bloque publicar tema para así tener una nueva instrucción de leer la humedad del suelo (Imagen 31).

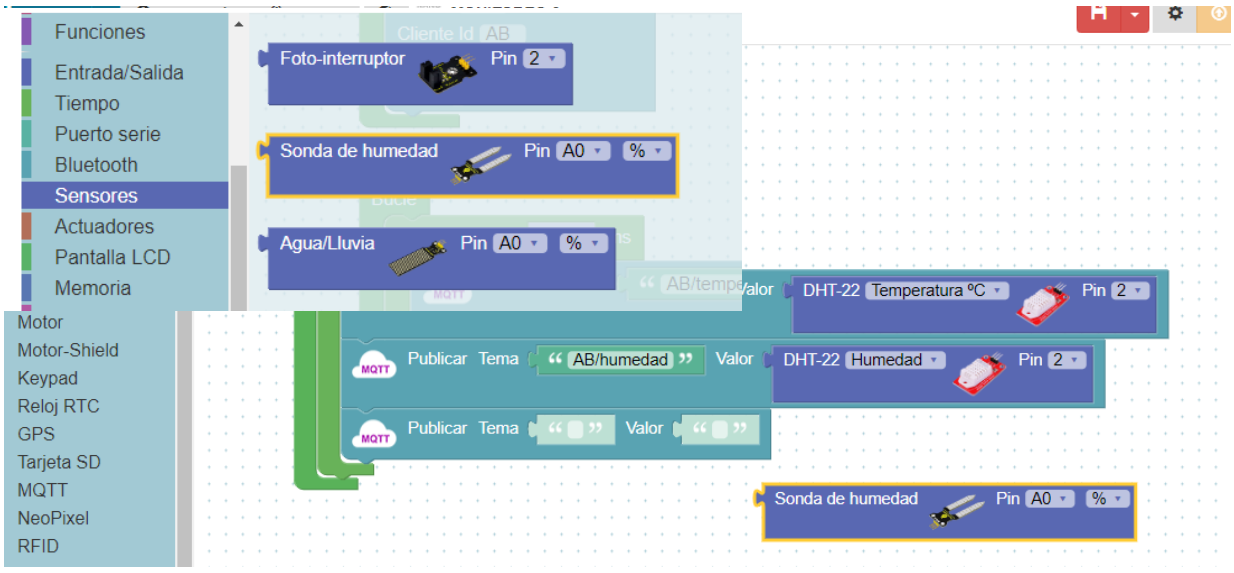


Imagen 31 Instrucción de lectura de humedad.

Con la instrucción creada necesitamos definir el nombre que llevara "*tierra1*" así como el pin en el cual se realizara la lectura en este caso "*Pin A8*" (Imagen 32).

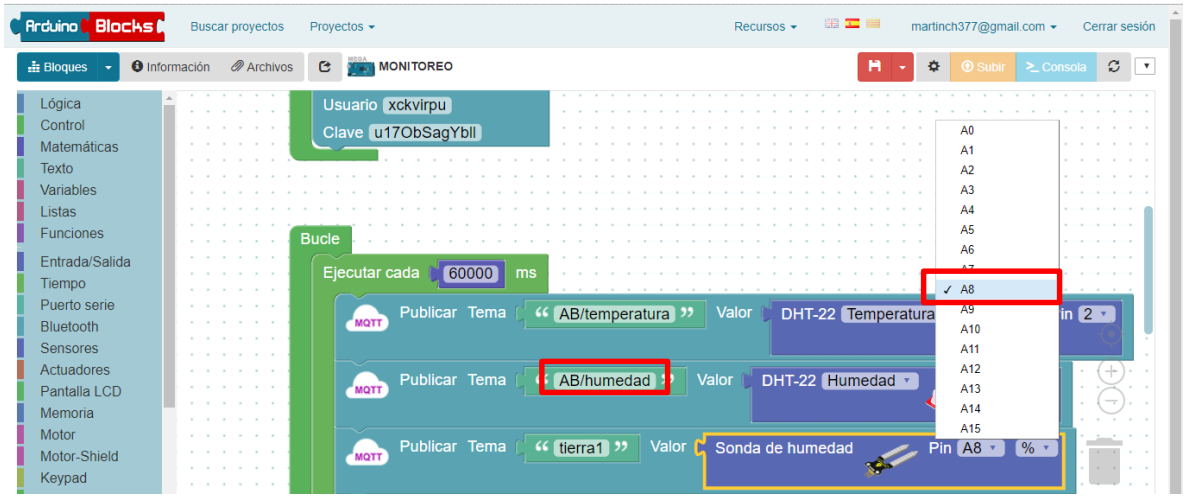


Imagen 32 Definición de la instrucción de lectura de humedad.

Después de haber creado la primera instrucción de lectura de humedad es necesario crear otras dos instrucciones para leer humedad. Para ello será necesario cambiar el nombre de la instrucción a "*tierra2*" y cambiar el pin de lectura al *pin A9* (Imagen 33).

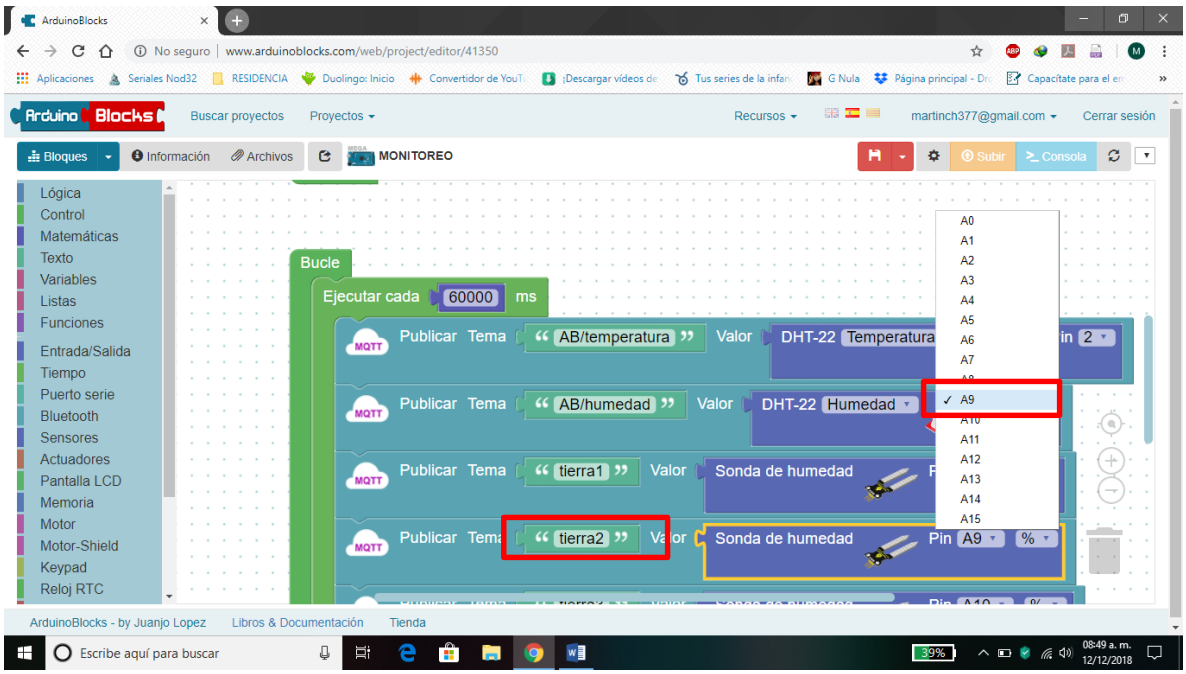


Imagen 33 Segunda instrucción de lectura de humedad.

Para la tercera instrucción de lectura de humedad del suelo, duplicamos la instrucción anterior, y así cambiar el nombre a "*tierra3*" de igual manera cambiar el pin de lectura al *pin A10* (Imagen 34).

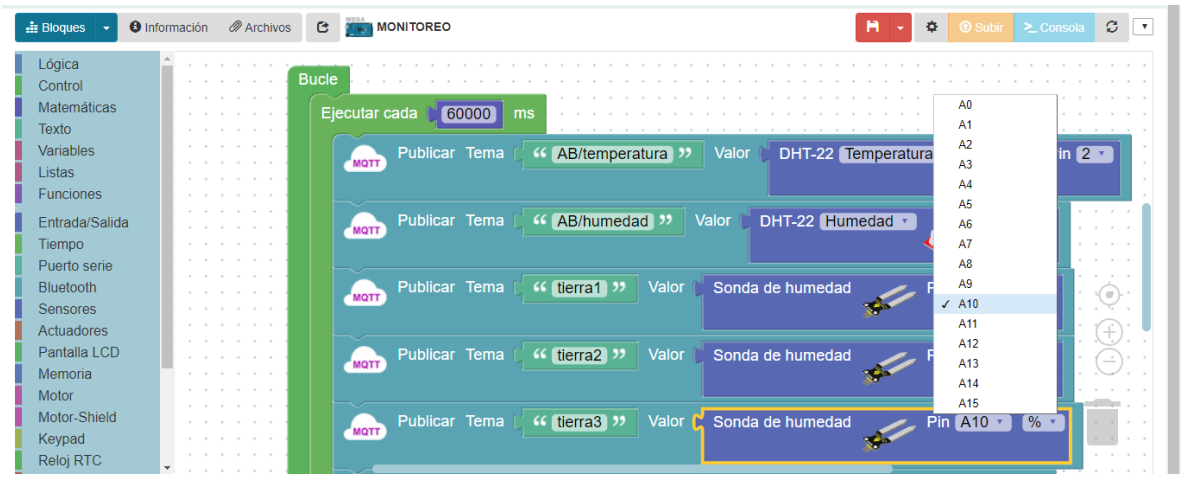


Imagen 34 Tercera instrucción para lectura de humedad del suelo.

Ya con las instrucciones de lectura de humedad de suelo, procedemos a crear la instrucción para la lectura de iluminación. Para ellos primero necesitamos ubicar el bloque del sensor LDR (Imagen 35).

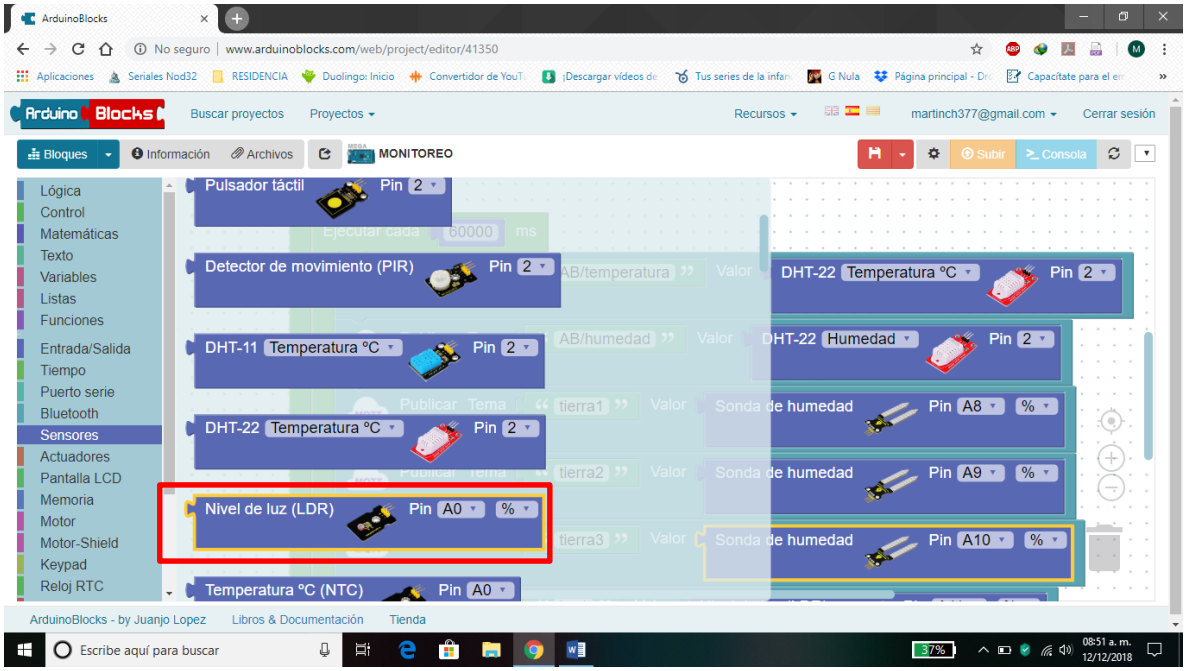


Imagen 35 Sensor LDR.

Ya ubicado el sensor serán necesarios definir el nombre y el pin de lectura en este caso "*luz1*" y el pin de lectura será el *A11* (Imagen 36).

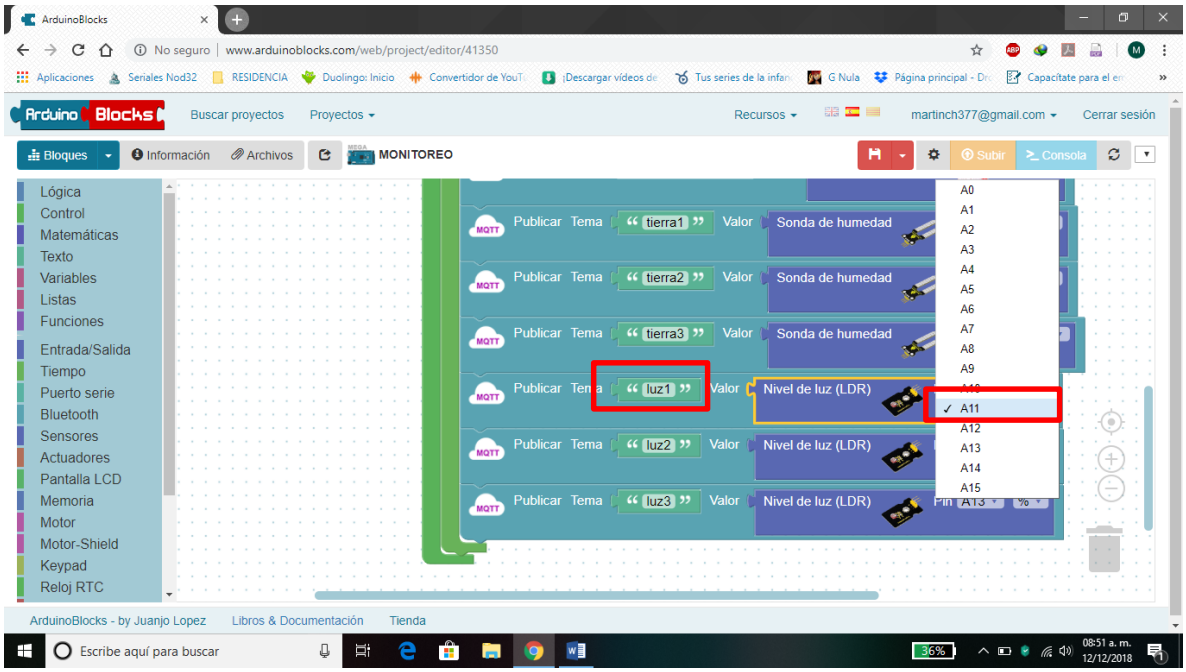


Imagen 36 Creación de la instrucción para lectura de iluminación.

Serán necesarios dos instrucciones más de lectura de iluminación, para ello duplicaremos dos veces la instrucción anterior, y así cambiar el nombre a "*luz2*" y "*luz3*" respectivamente. Así como cambiar los pines de lecturas a pin *A12* y *A13* respectivamente (Imagen 37).

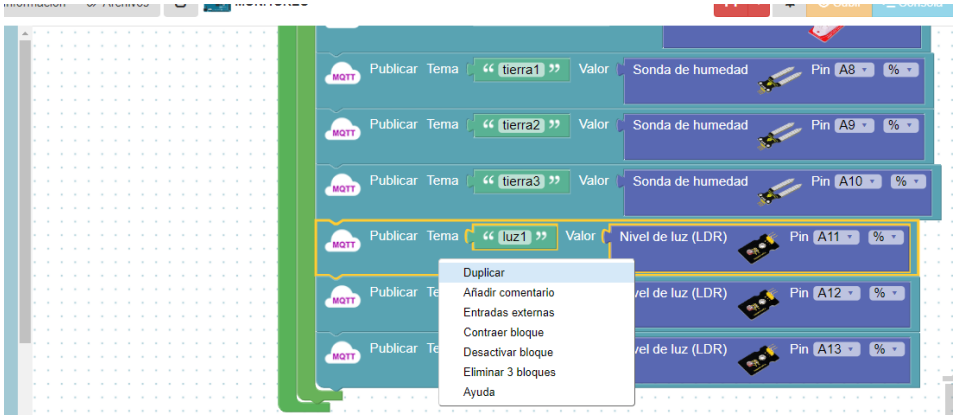


Imagen 37 Instrucciones para lectura de iluminación.

Ahora para verificar que el programa compila bien. Es necesario ejecutar "*ArduinoBlocks-Connector*" para poder realizar la conexión entre la placa arduino y la ArduinoBlocks para poder subir el código a la placa Arduino

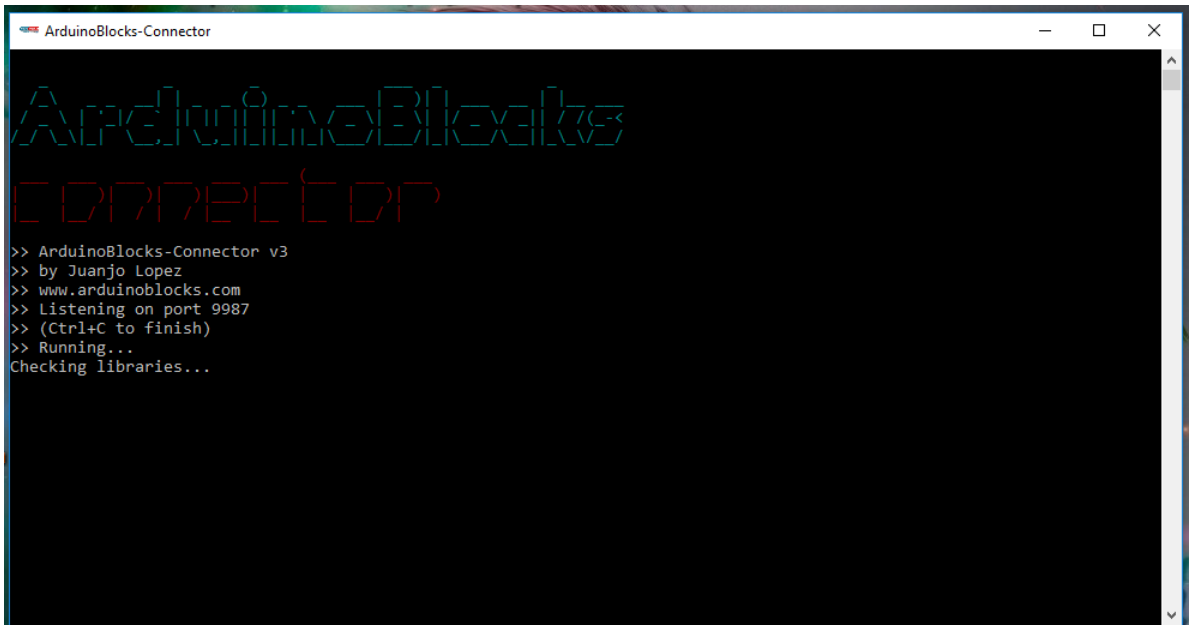


Imagen 38 Inicio de ArduinoBlocks-Connector.

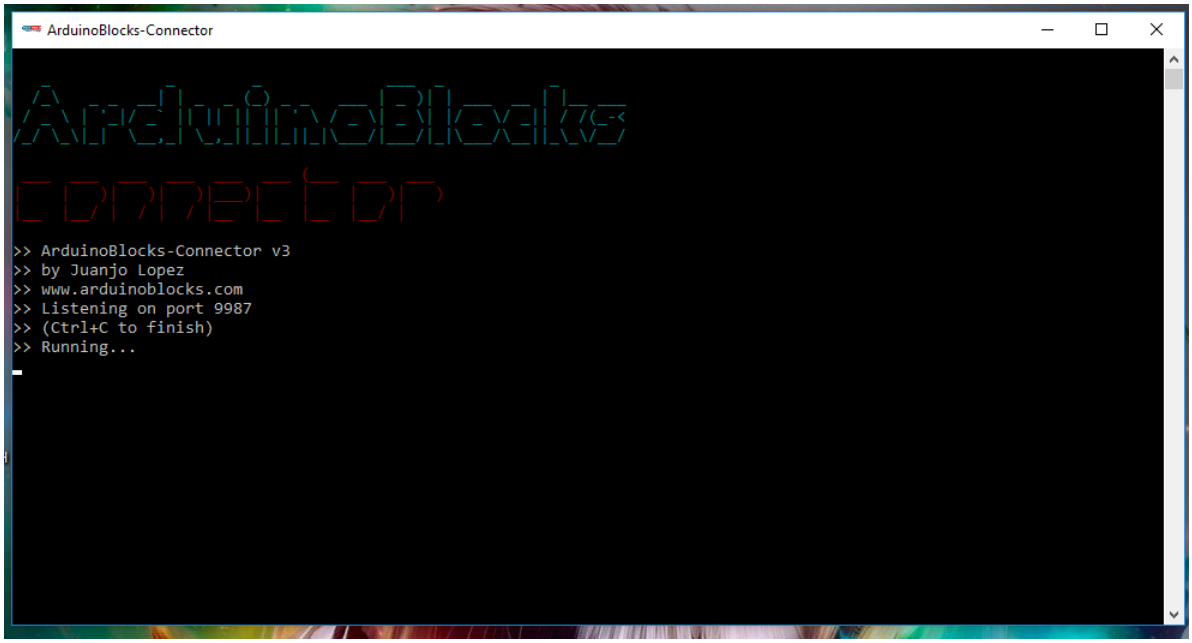


Imagen 39 ArduinoBlocks-Connector corriendo y listo para la conexión.

En la ventana del proyecto podemos ver que *ArduinoBlocks* detecto el puerto en el que está conectado en el COM5 (Imagen 40). Ya detectado el puerto, se habilita la opción subir, lo cual pasara el código a la placa (Imagen 41).

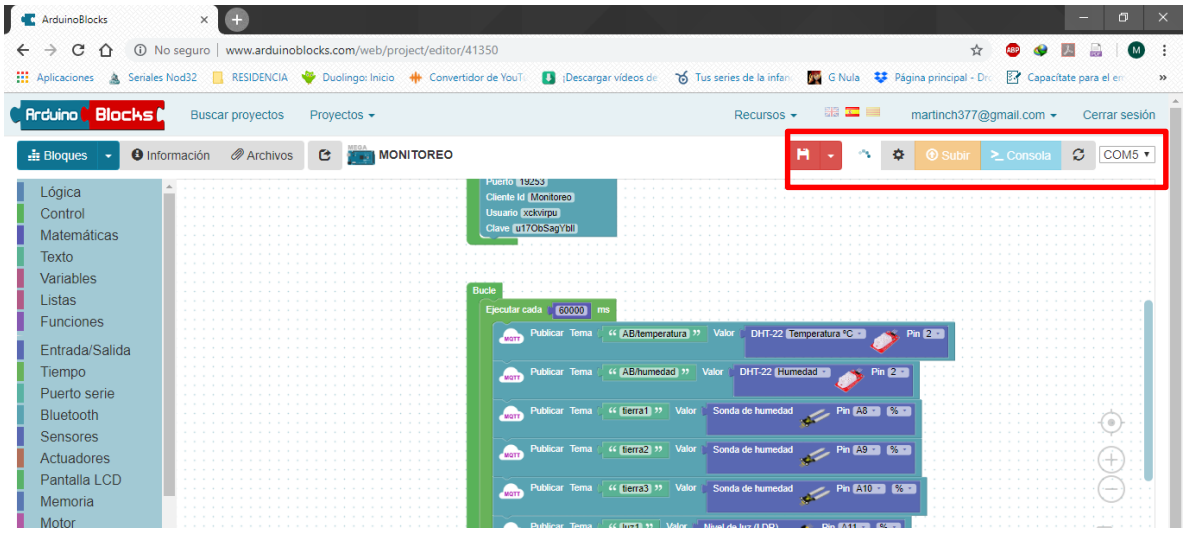


Imagen 40 Detección del puerto donde se conectó Arduino.

Cuando el código compilo y subió correctamente aparecerá el siguiente mensaje (Imagen 41).

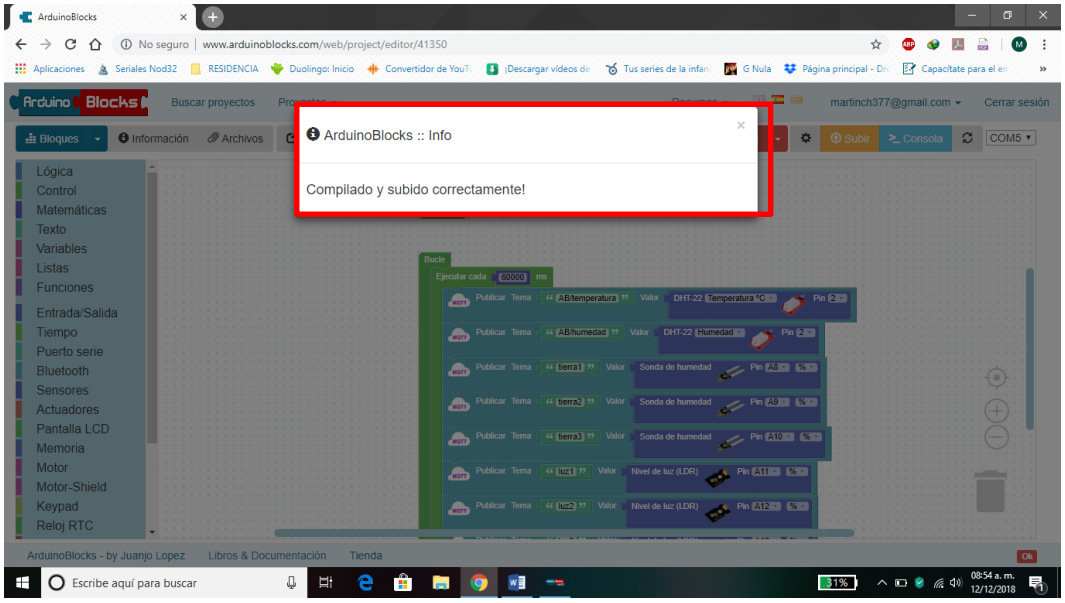


Imagen 41 Subida del código a la placa.

Creación del servidor que alojara los datos.

Para crear el servidor, ingresamos a *https://customer.cloudmqtt.com* e iniciamos sesión en este caso con la cuenta de Google (Imagen 42).

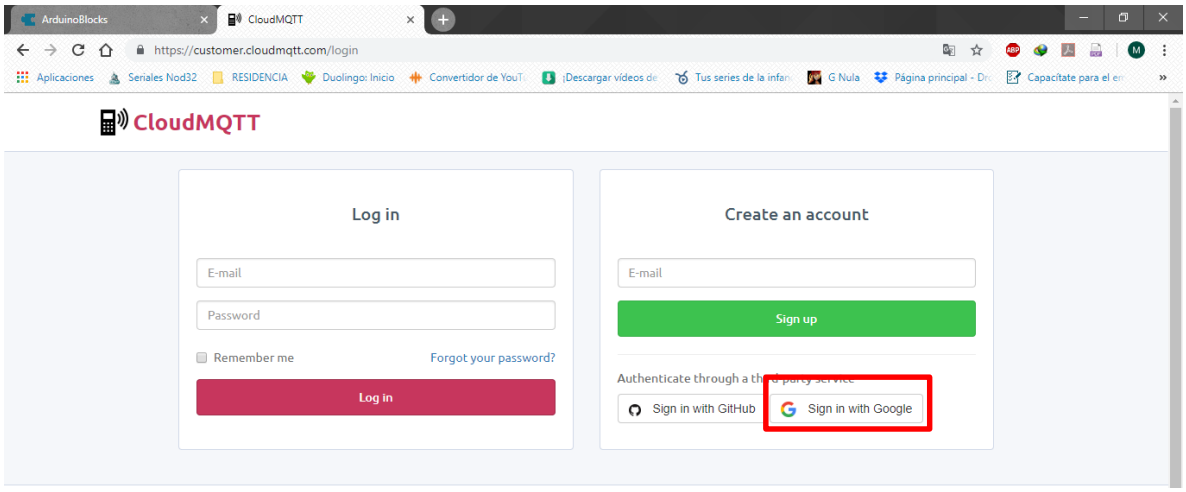


Imagen 42 CloudMQTT.

Ya después de haber iniciado sesión, se creará una nueva instancia la cual alojará los datos de las lecturas de los sensores

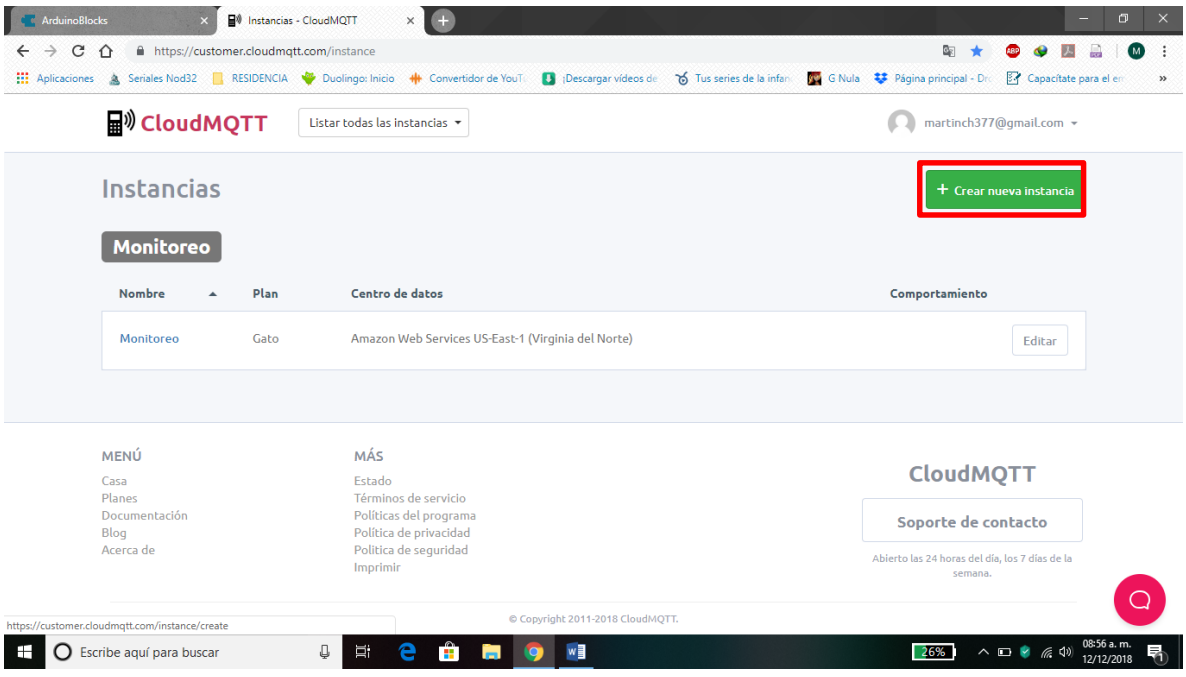


Imagen 43 Ventana principal del servidor.

Ahora bien, es necesario definir el nombre de la instancia que tendrá en el servidor, así como el plan en el que se trabajará en este caso el plan gratis (Imagen 44).

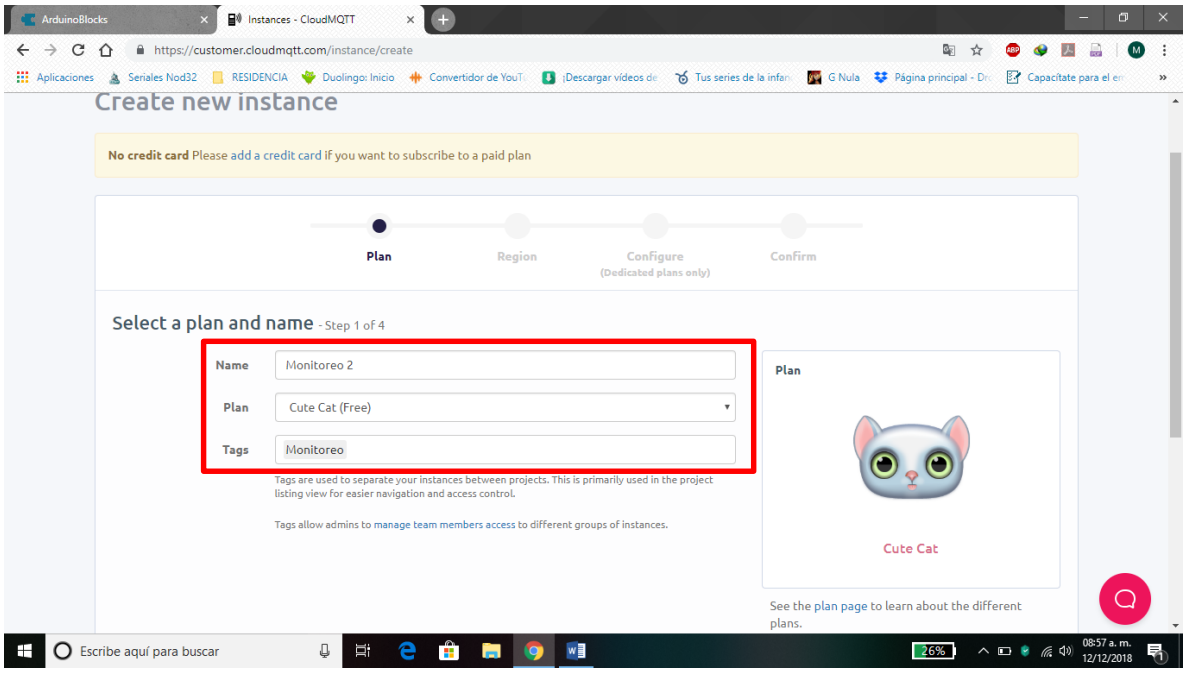


Imagen 44 Definición de la instancia.

Con los parámetros ya definidos, vamos a seleccionar la región en el cual se alojará el servidor

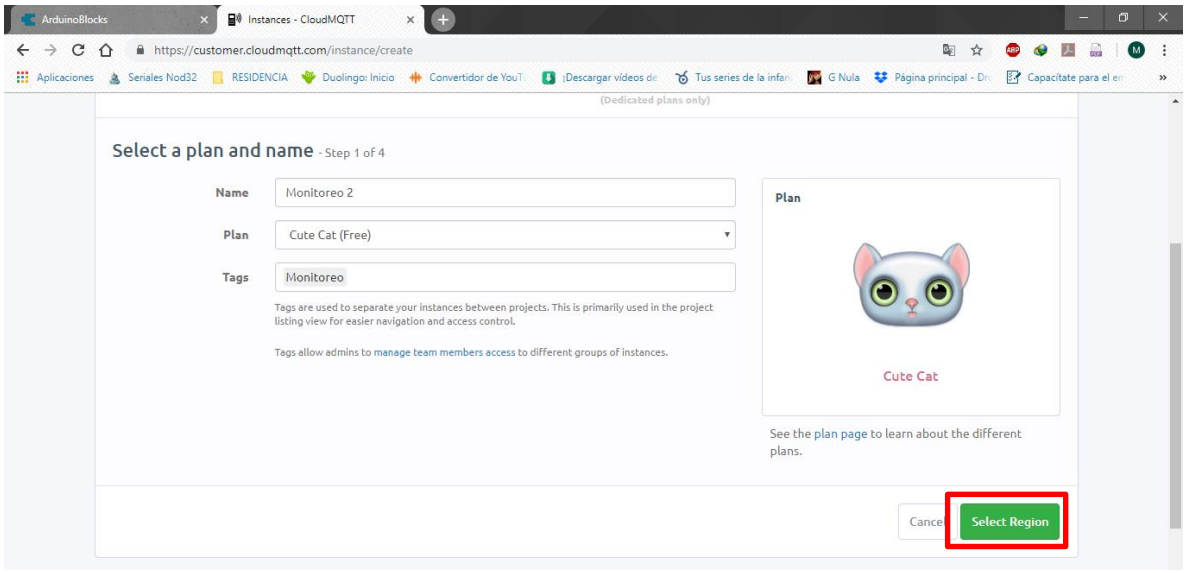


Imagen 45 Definir región.

Ahora ya en la sección de seleccionar región, elegiremos *US-East-1(Northern Virginia)*, y presionamos *Review* (Imagen 46).

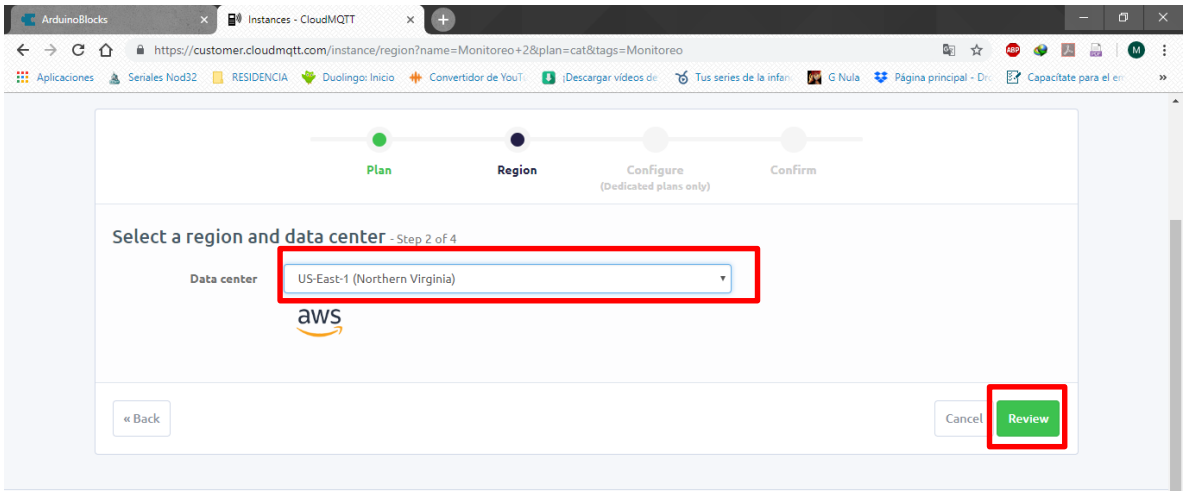


Imagen 46 Selección de la región.

Ahora ya con los parámetros bien definidos, presionamos *crear la instancia* (Imagen 47).

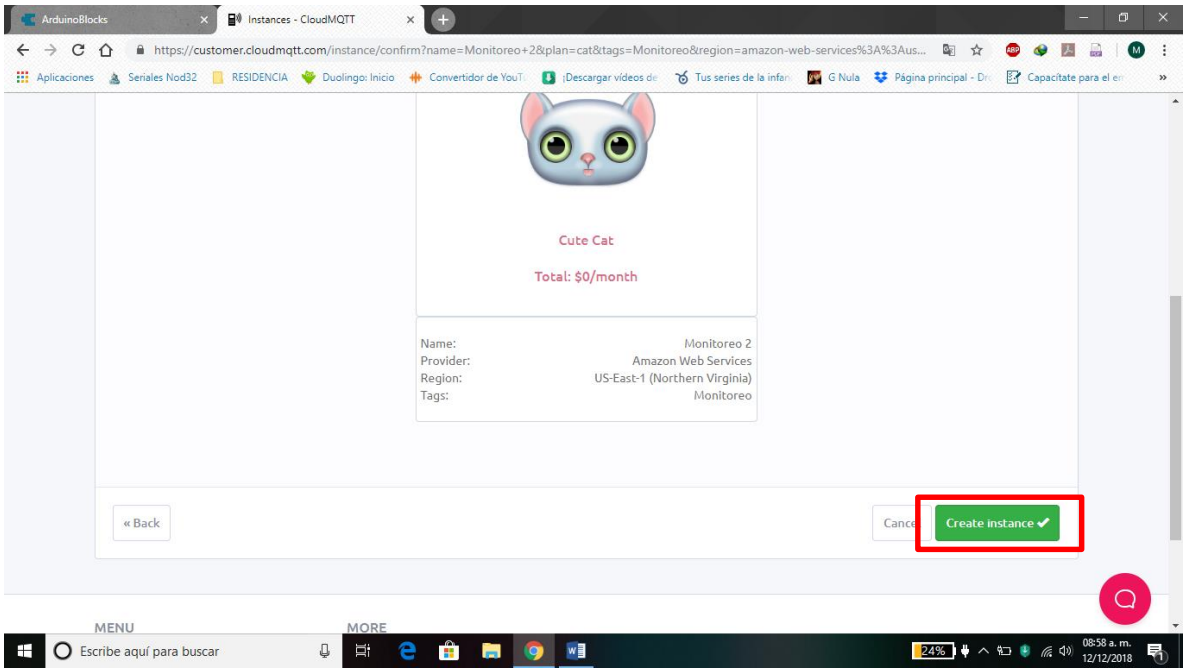


Imagen 47 Confirmación de creación.

Ahora que tenemos la instancia creada, podemos ver los detalles del servidor, así como crear usuarios del mismo (Imagen 48).

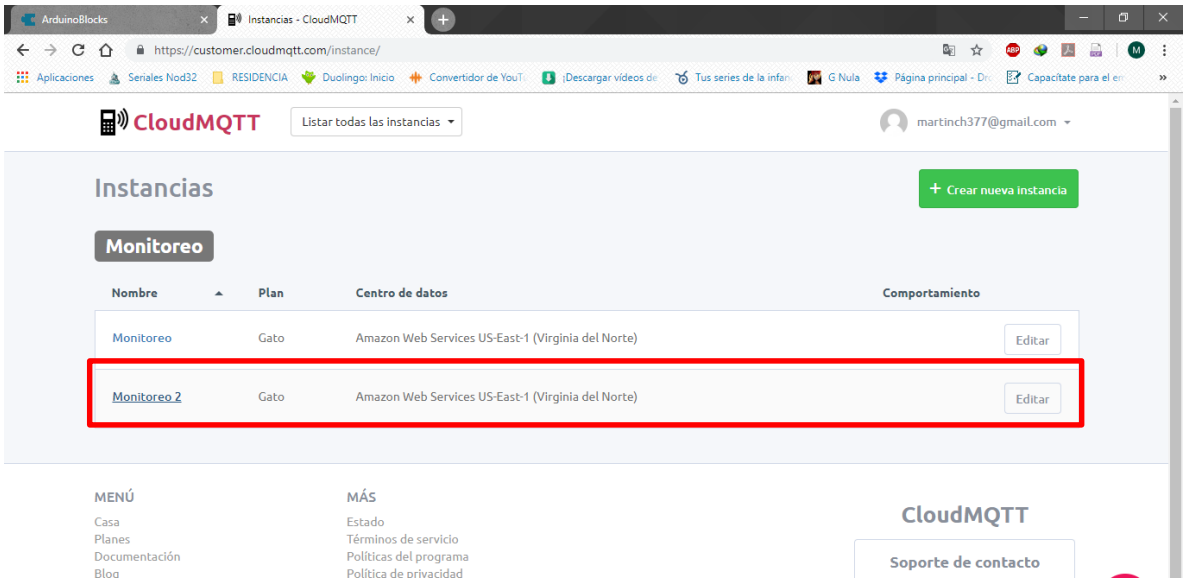


Imagen 48 Instancia creada en el servidor.

En la siguiente venta se pueden visualizar los detalles de la instancia los cuales serán útiles para la conexión al servidor (Imagen 49).

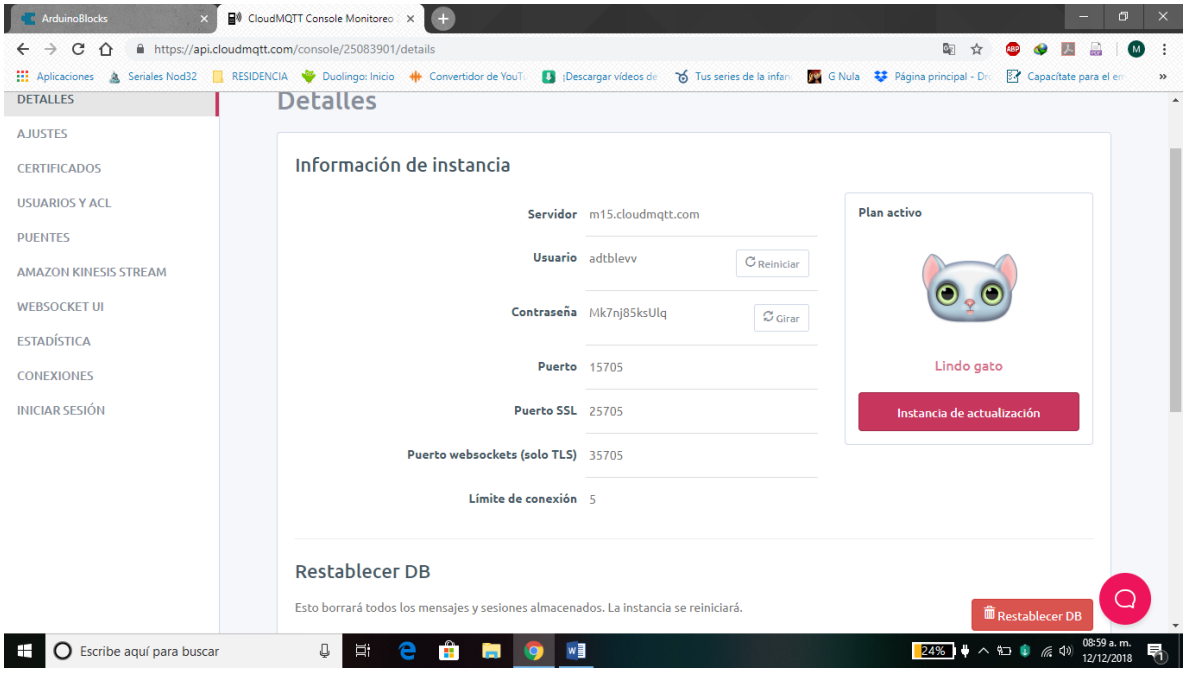


Imagen 49 Detalles del servidor.

En esta sección crearemos un usuario para el servidor el cual enviara los datos a la instancia alojada en el servidor (Imagen 50).

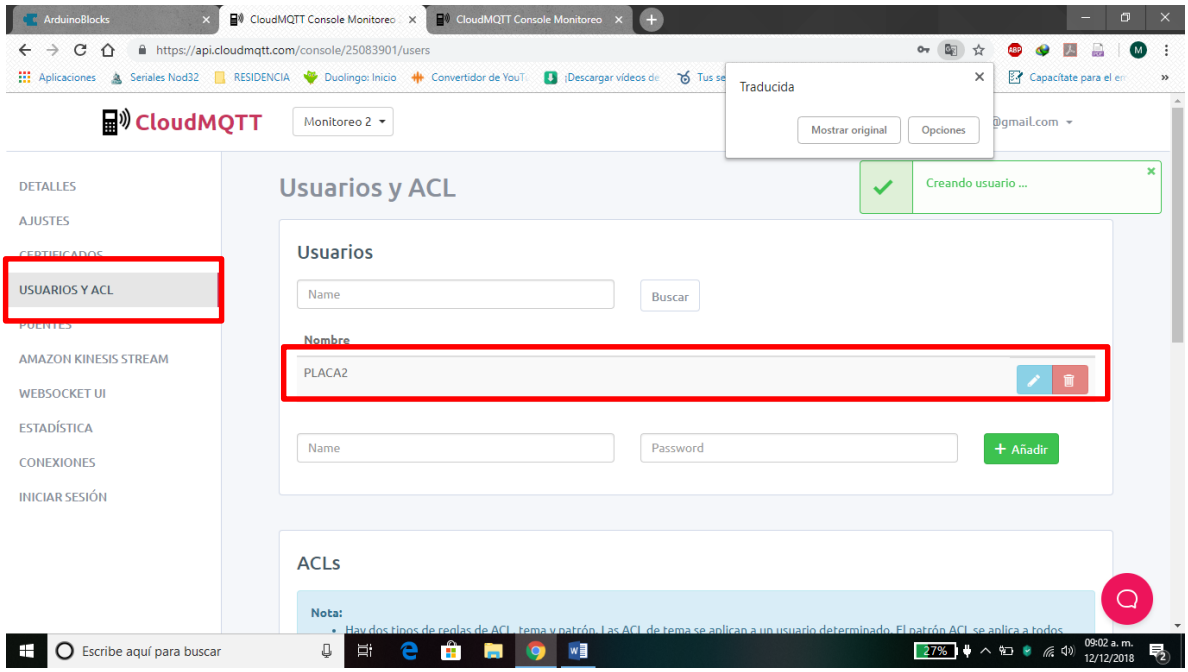


Imagen 50 Creación de usuario en el servidor.

Además, también crearemos un *ACLs*, el cual permitirá la lectura y escritura de los valores de lectura (Imagen 51).

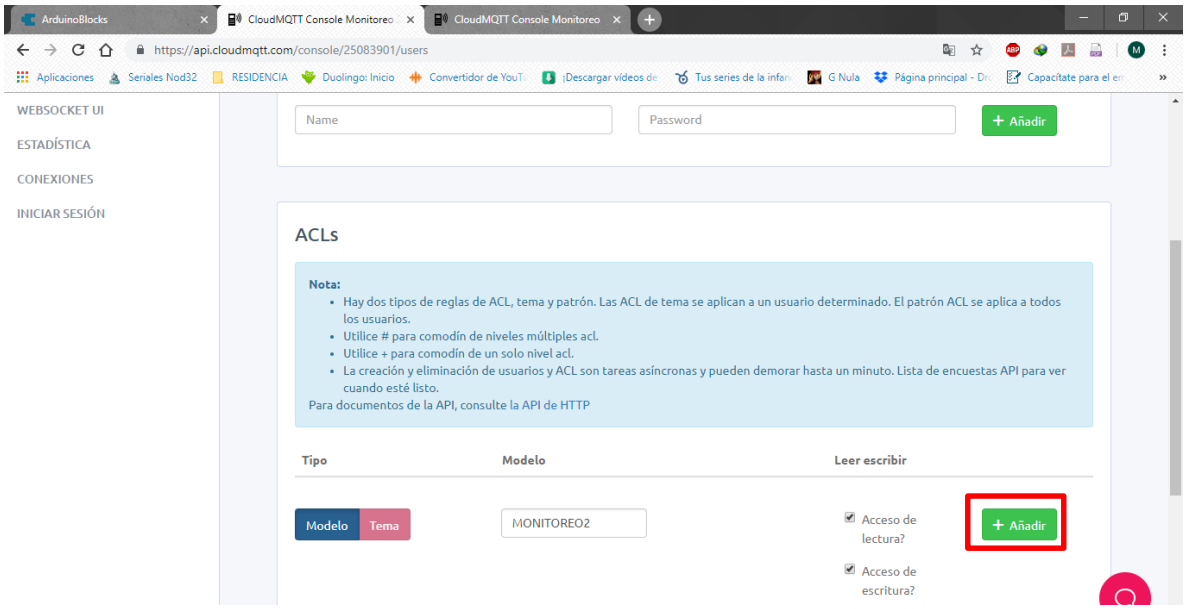


Imagen 51 Creación de ACLs.

Configuración entre Servidor y Arduino.

Para la configuración entre el servidor y arduino, es necesario ingresa al proyecto que se creó en *ArduinoBlocks*, para asignar los valores del servidor al bloque de *MQTT* (Imagen 52).

Imagen 52 Proyecto en ArduinoBlocks.ç

En la siguiente ventana se observan los detalles de la instancia, del cual obtendremos la dirección del servidor para poder ingresar al bloque *MQTT* (Imagen 53).

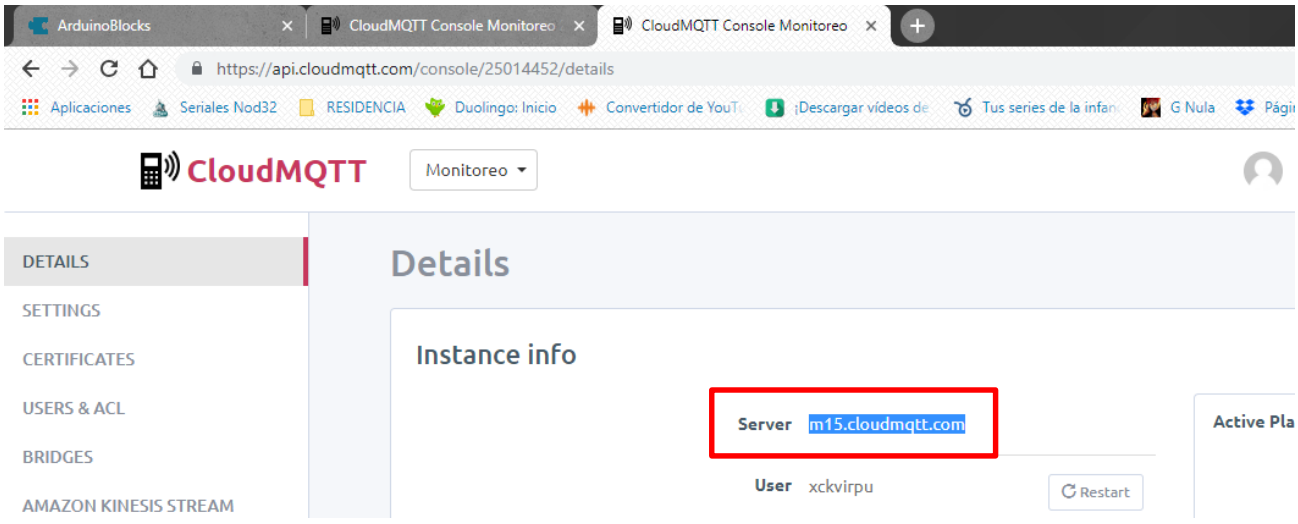


Imagen 53 Dirección del servidor.

Ahora que ya tenemos la dirección del servidor, se pondrá en el apartado "*Broker*" para poder direccionar al servidor desde arduino (Imagen 54).

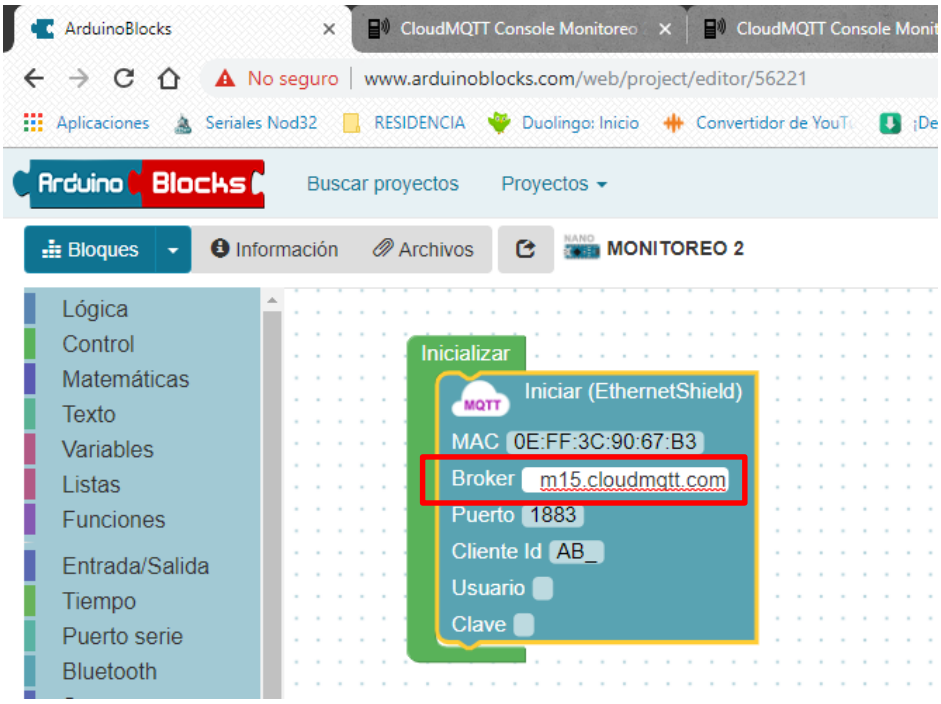


Imagen 54 Definición de la dirección del servidor en Arduino.

Con la dirección definida, continua definir el puerto en el cual será la conexión en el servidor y Arduino

Imagen 55 Puerto de conexión.

Ahora se ingresa el puerto de conexión al apartado puerto del bloque MQTT (Imagen 56).

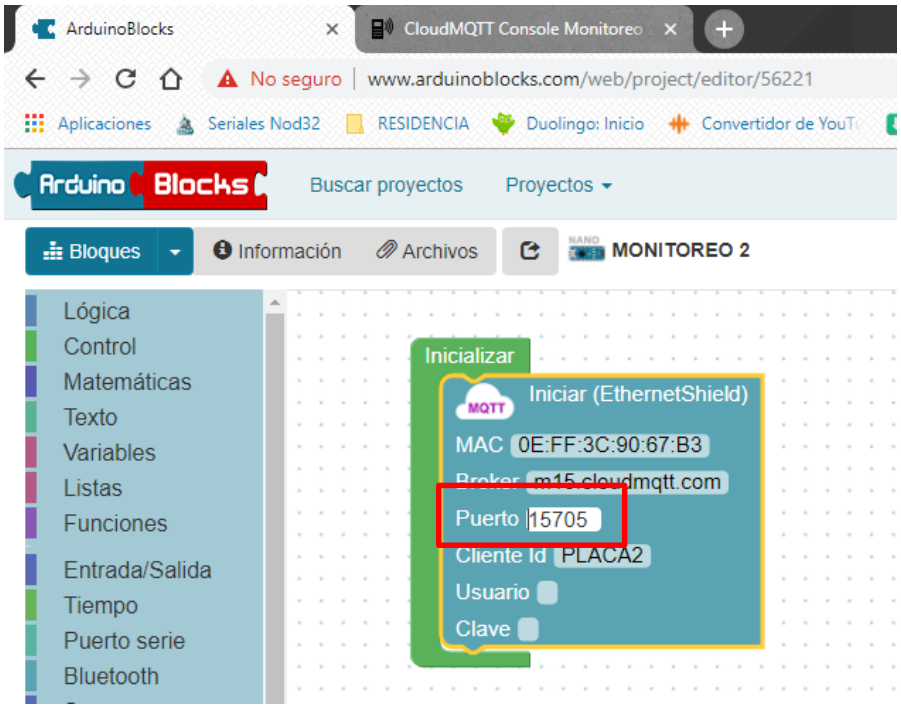


Imagen 56 Definición del puerto de conexión.

A continuación, en la instancia obtendremos el usuario mediante el cual tendremos acceso para lectura y escritura en el servidor (Imagen 57).

Imagen 57 Usuario del servidor.

Ahora se ingresa el nombre de usuario al apartado usuario del bloque MQTT (Imagen 56).

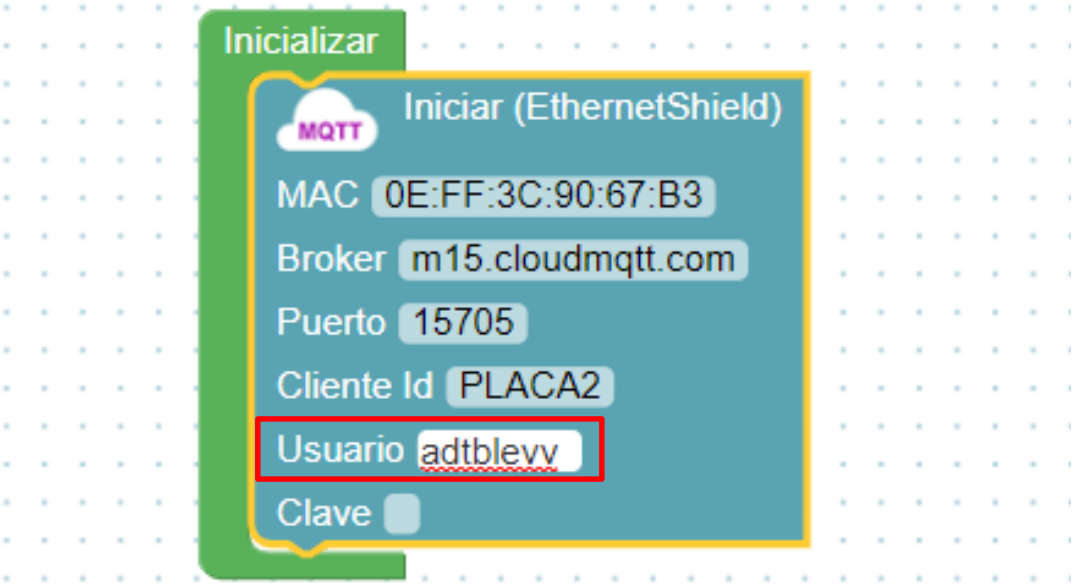


Imagen 58 Ingreso del nombre de usuario para el acceso desde Arduino.

Ahora obtenemos la contraseña para validar el acceso a lectura y escritura en el servidor (Imagen 59).

Imagen 59 Contraseña de acceso al servidor.

Ingresamos la contraseña en el apartado clave del bloque MQTT. Ahora ya con todos los parámetros definidos procedemos a subir el código nuevamente a la placa Arduino Mega (Imagen 60).

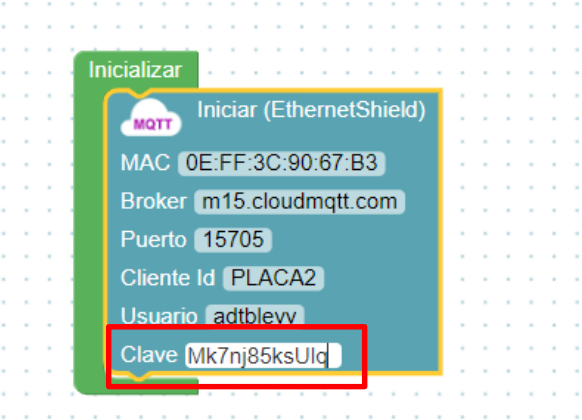


Imagen 60 Ingreso de la contraseña.

Ahora podremos ver el código del programa desde el IDE de Arduino (Imagen 61-62).

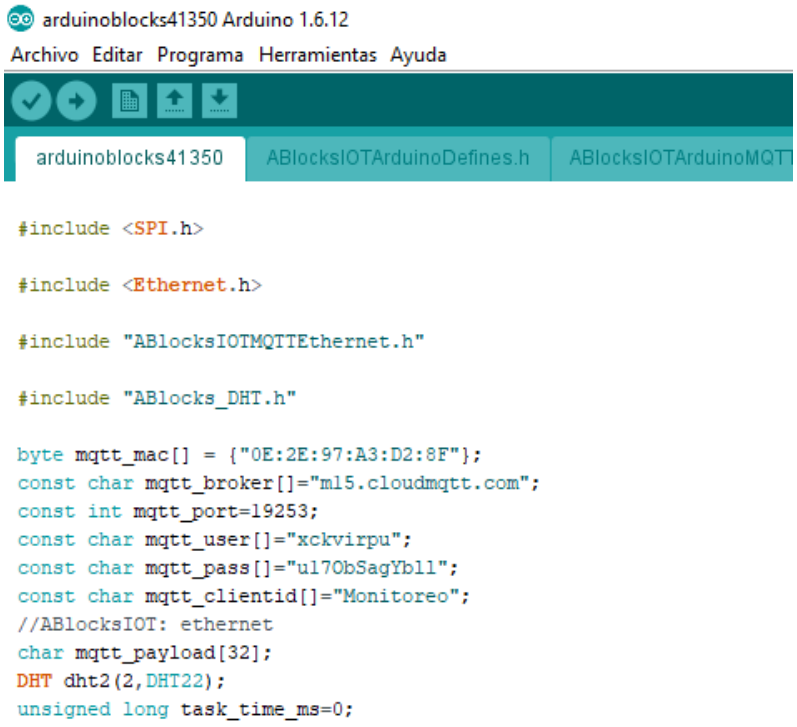


Imagen 61 Arduino IDE.

```
double mqtt_payload2double(unsigned char *_payload, int _length){
  int i;
  for (i = 0; i<_length && i<32; i++){
    mqtt_payload[i] = _payload[i];
  }
  mqtt_payload[i] = 0;
  return atof(mqtt_payload);
}

String mqtt_payload2string(unsigned char *_payload, int _length){
  int i;
  for (i = 0; i<_length && i<32; i++){
    mqtt_payload[i] = _payload[i];
  }
  mqtt_payload[i] = 0;
  return String(mqtt_payload);
}
void mqtt_callback(char* _topic, unsigned char* _payload, unsigned int _payloadlength){
  double v=mqtt_payload2double(_payload,_payloadlength);
  String vt=mqtt_payload2string(_payload,_payloadlength);
}

void mqtt_subscribe(){
}


void setup()
{
    ABlocksIOT.begin(mqtt_broker,mqtt_port, mqtt_user,mqtt_pass, mqtt_clientid, mqtt_mac, mqtt_callback, mqtt_subscribe);
pinMode(2, INPUT);
dht2.begin();
pinMode(A8, INPUT);
pinMode(A9, INPUT);
pinMode(A10, INPUT);
pinMode(A11, INPUT);
pinMode(A12, INPUT);
pinMode(A13, INPUT);
}


void loop()
{
  ABlocksIOT.loop();
    if((millis()-task_time_ms)>=60000){
      task_time_ms=millis();
      ABlocksIOT.Publish(String(String("AB/temperatura")), String(dht2.readTemperature()));
      ABlocksIOT.Publish(String(String("AB/humedad")), String(dht2.readHumidity()));
      ABlocksIOT.Publish(String(String("tierra1")), String(map(analogRead(A8),0,1023,0,100)));
      ABlocksIOT.Publish(String(String("tierra2")), String(map(analogRead(A9),0,1023,0,100)));
      ABlocksIOT.Publish(String(String("tierra3")), String(map(analogRead(A10),0,1023,0,100)));
      ABlocksIOT.Publish(String(String("luz1")), String(map(analogRead(A11),0,1023,0,100)));
      ABlocksIOT.Publish(String(String("luz2")), String(map(analogRead(A12),0,1023,0,100)));
      ABlocksIOT.Publish(String(String("luz3")), String(map(analogRead(A13),0,1023,0,100)));
    }

}
```

Imagen 62 Continuación del código en Arduino IDE.

Configuración de un cliente “Android”.

Para iniciar la configuración entre el servidor y un dispositivo Android, primero es necesario ingresar a *PlayStore*, para descargar la app oficial del servidor (Imagen 63).

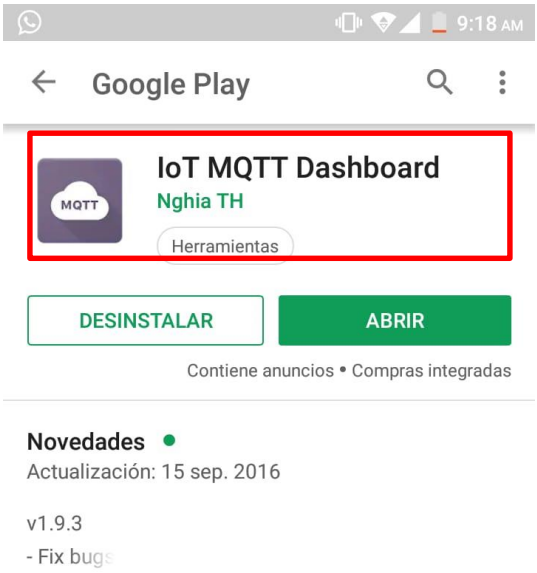


Imagen 63 App MQTT.

Ya con la app instalada, en la ventana principal tenemos lo siguiente en el cual presionaremos el icono + para agregar los datos del servidor (Imagen 64).

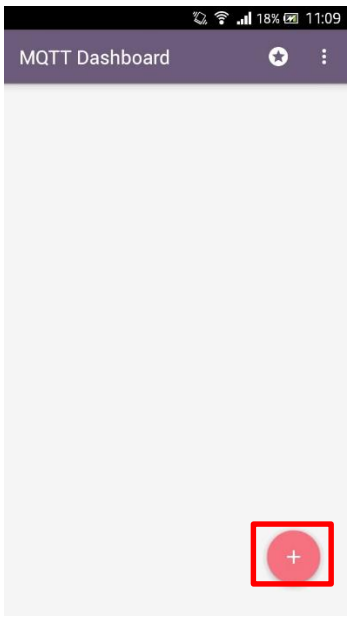


Imagen 64 Ventana principal de la App.

En la siguiente ventana agregamos los parámetros del servidor, para poder ingresar a los datos del servidor y presionamos *save* para guardar los datos (Imagen 65).

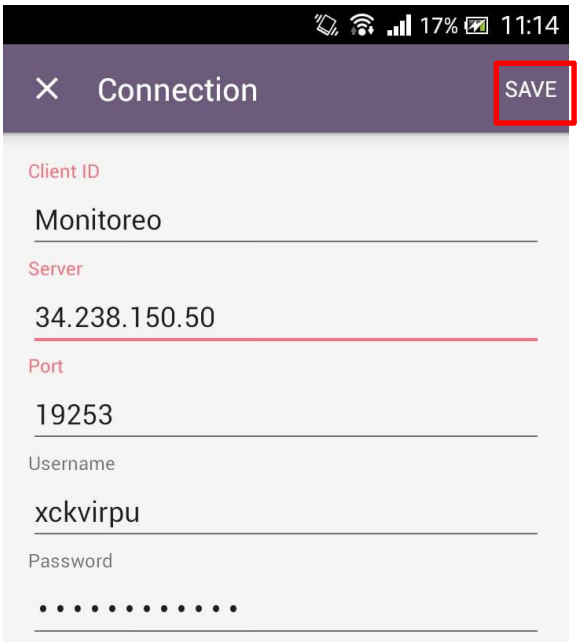


Imagen 65 Definición de los parámetros de conexión.

Esta es la ventana donde se verán las lecturas realizadas, pero primero hay que suscribirse a los parámetros que se definición como lecturas (Imagen 66).

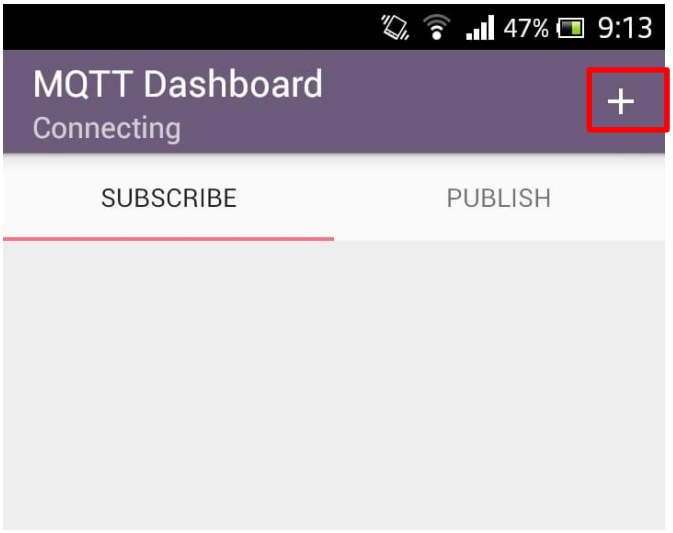


Imagen 66 Ventana principal del servidor visto desde Android.

Como primera suscripción agregaremos el nombre con el cual se visualizará, al igual que el nombre de la instrucción de lectura que queremos ver en este caso *AB/humedad*, además de elegir la unidad con la que se visualizará (Imagen 67).

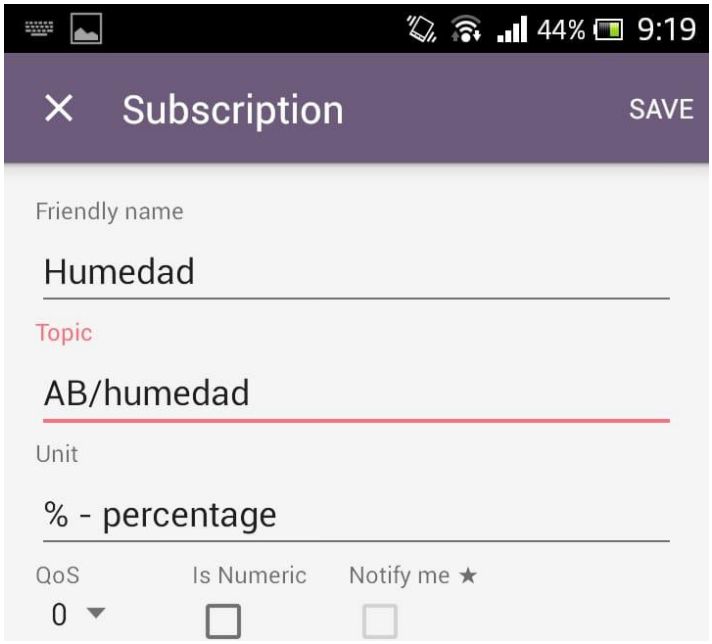


Imagen 67 Suscripción a Humedad.

Ahora lo mismo se realizará con la instrucción *Temperatura* (Imagen 68).

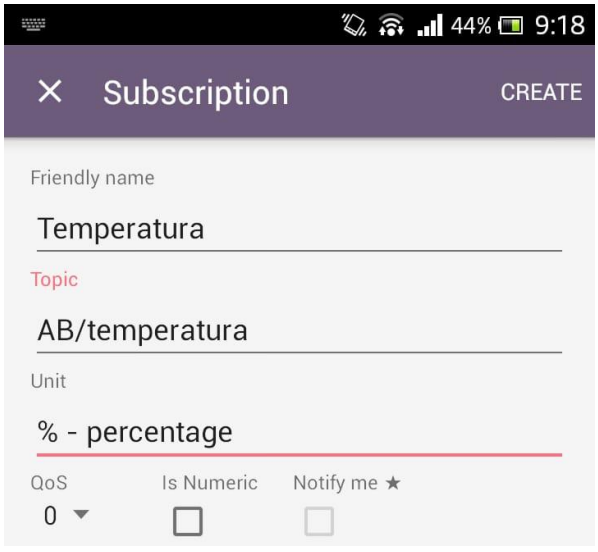


Imagen 68 Suscripción a Temperatura.

Eso se realizará con todas las instrucciones (Imagen 69).

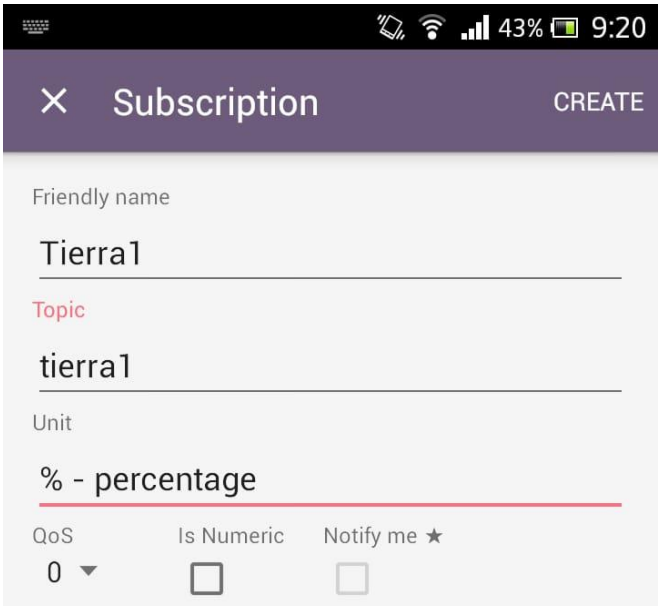


Imagen 69 Suscripción a Tierra 1.

Esta será la ventana donde se visualizará las lecturas obtenidas (Imagen 70).

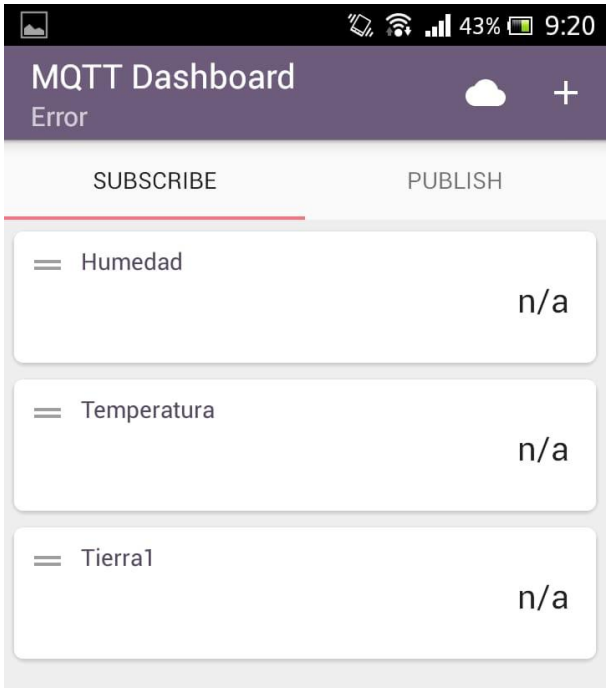


Imagen 70 Vista de las suscripciones.

Vista desde la App de todas las *suscripciones* (Imagen 71).

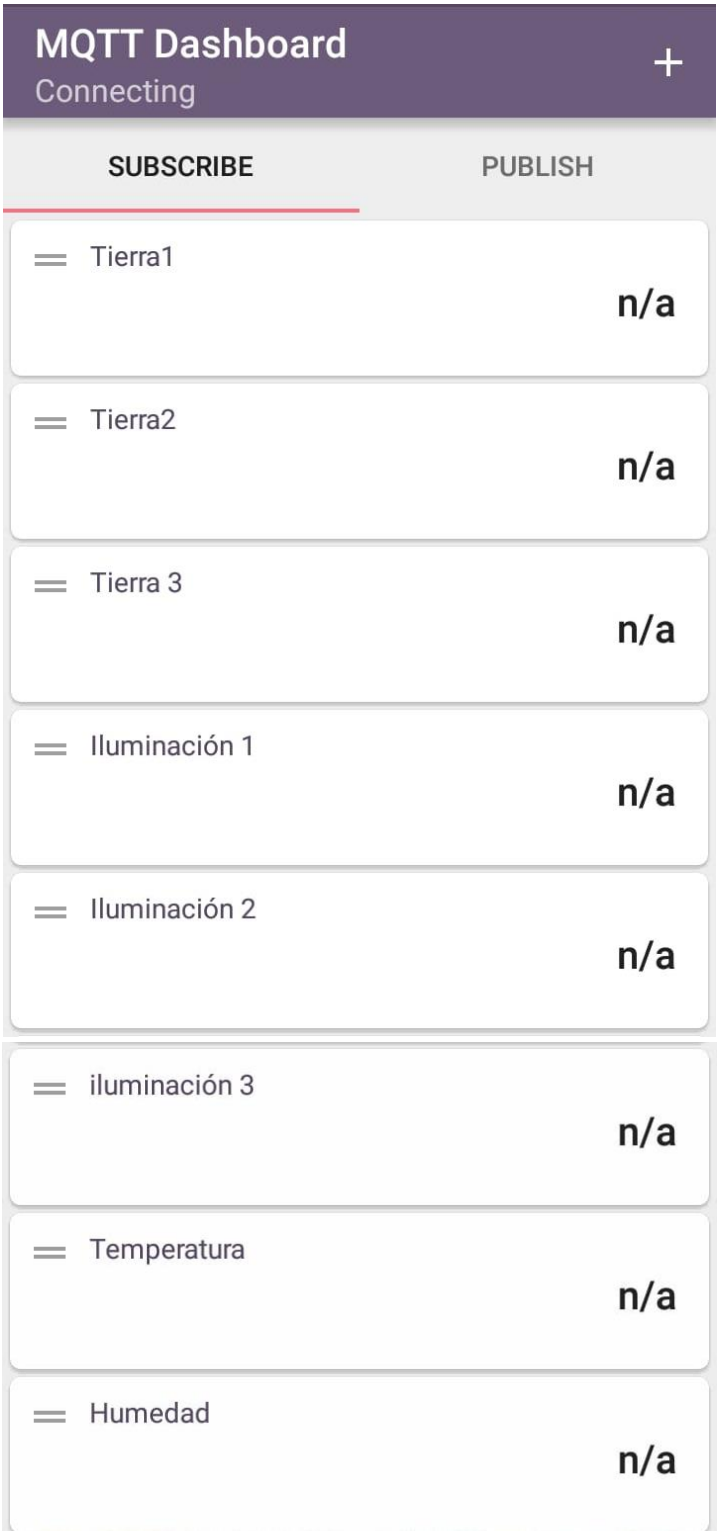


Imagen 71 Vista de todas las suscripciones.

Diseño de la fuente de distribución de voltaje.

Como primera parte del diseño en *Proteus*, elegimos los componentes que necesitamos en *Schematic* Capture (Imagen 72).

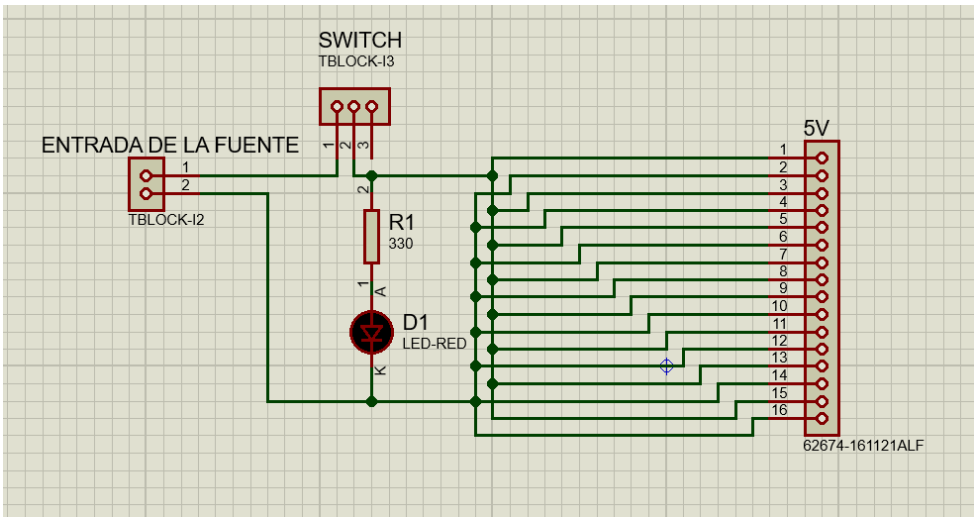


Imagen 72 Diseño en Proteus.

Ahora que ya tenemos todos los componentes nos dirigimos a *PCB Layout* para realizar el diseño que será traspasado a la placa fenólica virgen (Imagen 73).

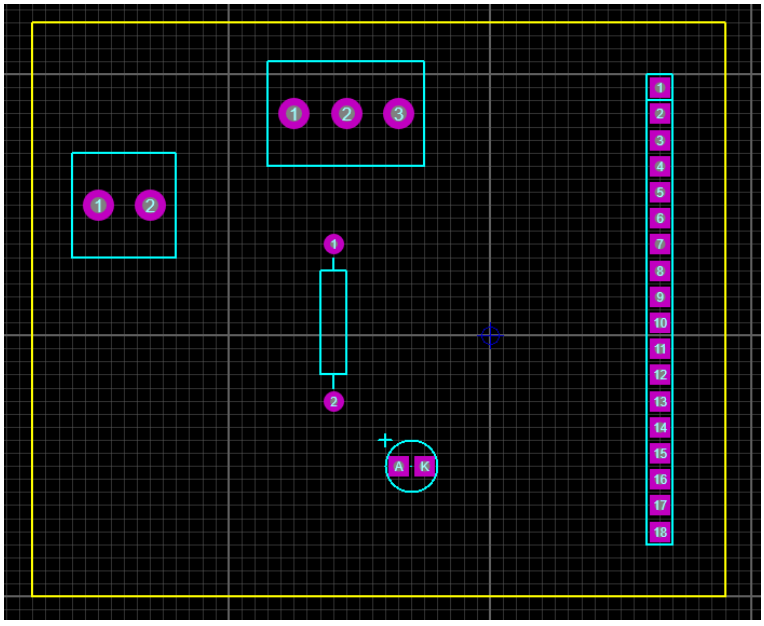


Imagen 73 Vista PCB Layout.

Ya con el diseño en *PCB* podemos *visualizar en 3D* como se vería la distribución de los componentes de lo que sería la fuente de distribución (Imagen 74).

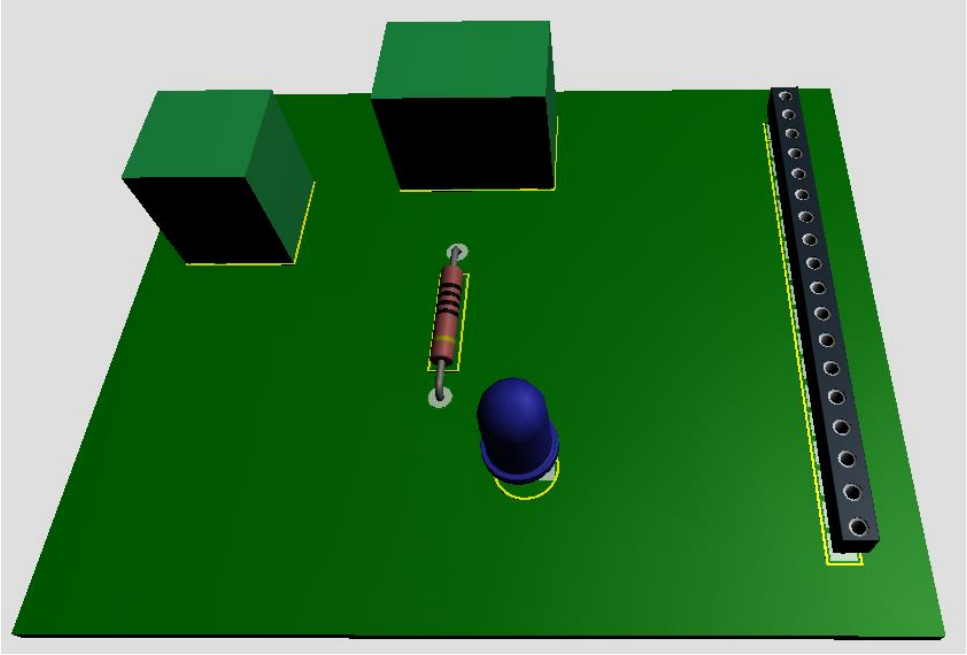


Imagen 74 3D Visualizer.

Conforme al diseño ya correctamente diseñado, procedemos convertir el PCB a PDF para poder realizar la transferencia del circuito a la placa fenólica, para ello es necesario realizar la impresión del PDF en impresora láser (Imagen 75).

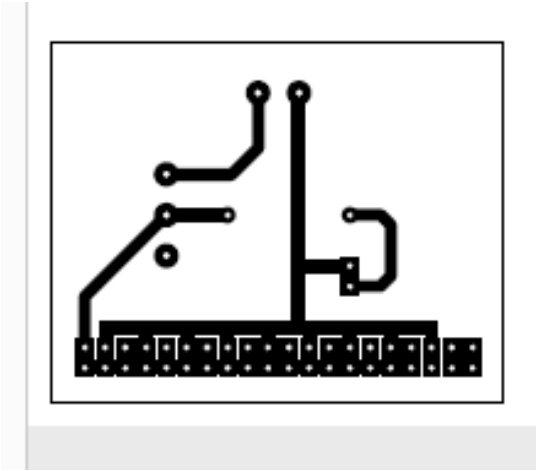


Imagen 75 Diseño en PDF.

Para realizar la transferencia del circuito desde la impresión realizada en laser, procedemos a colocar de manera correcta sobre la placa fenólica para posteriormente planchar el circuito para realizar la transferencia (Imagen 76).

Imagen 76 Planchado de la placa.

Con la transferencia realizada correctamente, colocamos la placa dentro de un recipiente con *Cloruro Férrico "$FeCl_3$"*, para retirar el recubrimiento de cobre sobrante (Imagen 77).

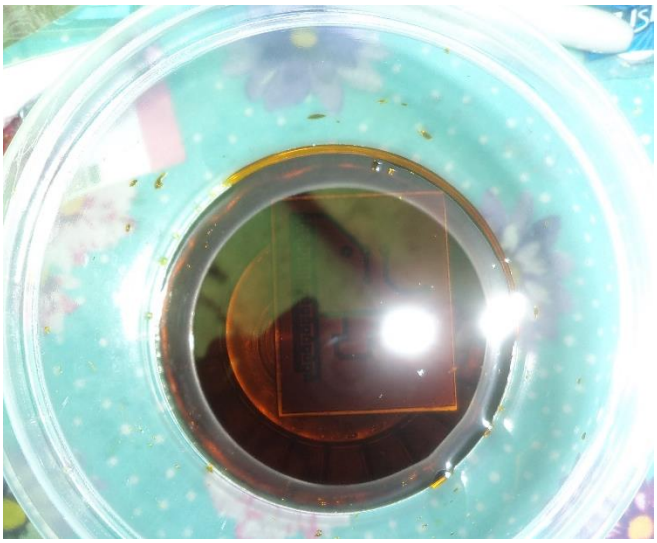


Imagen 77 Placa introducida a Cloruro Férrico.

Con el cobre sobrante retirado, la placa quedo de la siguiente manera. Ahora para retirar el recubrimiento de tóner aplicaremos acetona para retirar el recubrimiento (Imagen 78).

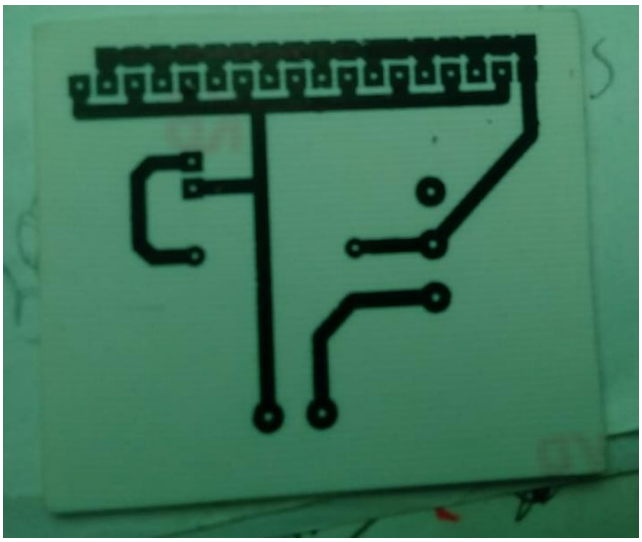


Imagen 78 Placa con recubrimiento de tóner.

Ya que retiramos el recubrimiento la placa se vera de la siguiente manera. Después de eso procedemos a realizar las perforaciones con la ayuda de un Dremel; de donde irán colocados los componentes de la placa (Imagen 79).

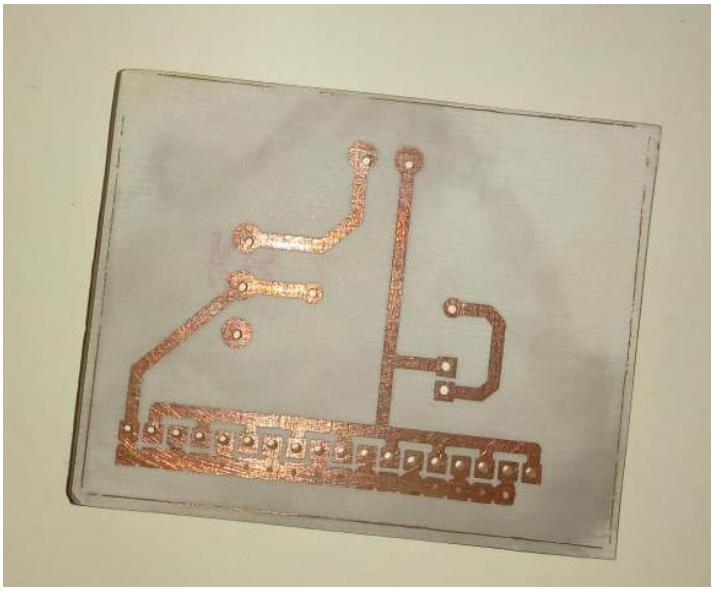


Imagen 79 Placa perforada.

Ya realizado todo lo anterior colocamos los componentes de manera correcta (Imagen 80).

Imagen 80 Componentes en la placa.

Con los componentes correctamente colocados, procedemos a realizar la soldadura de los componentes (Imagen 81). Y así tenemos la placa terminada (Imagen 82).

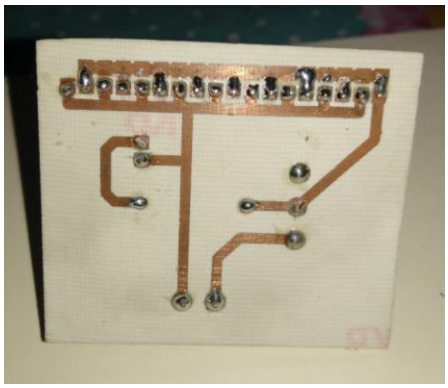
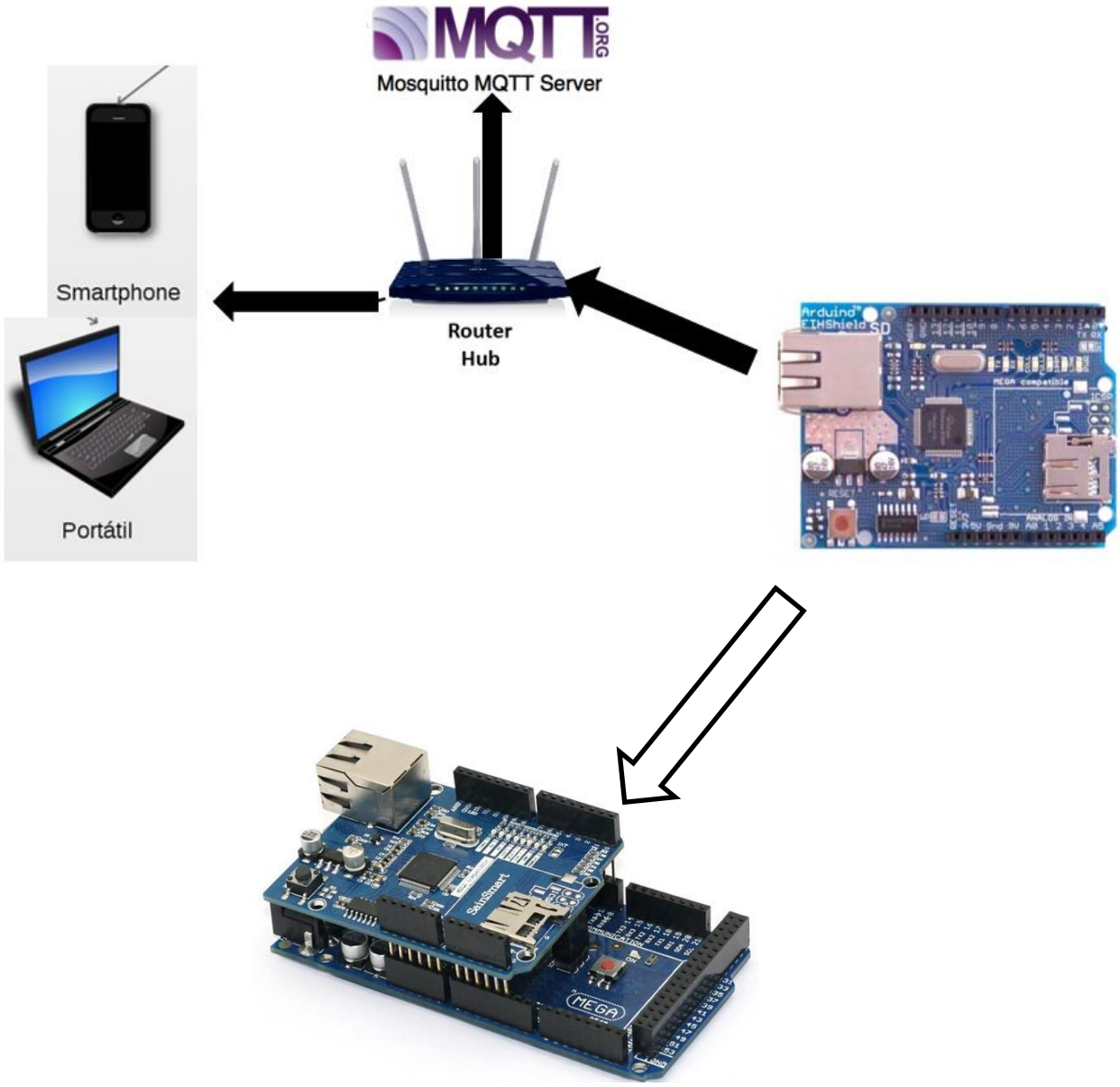
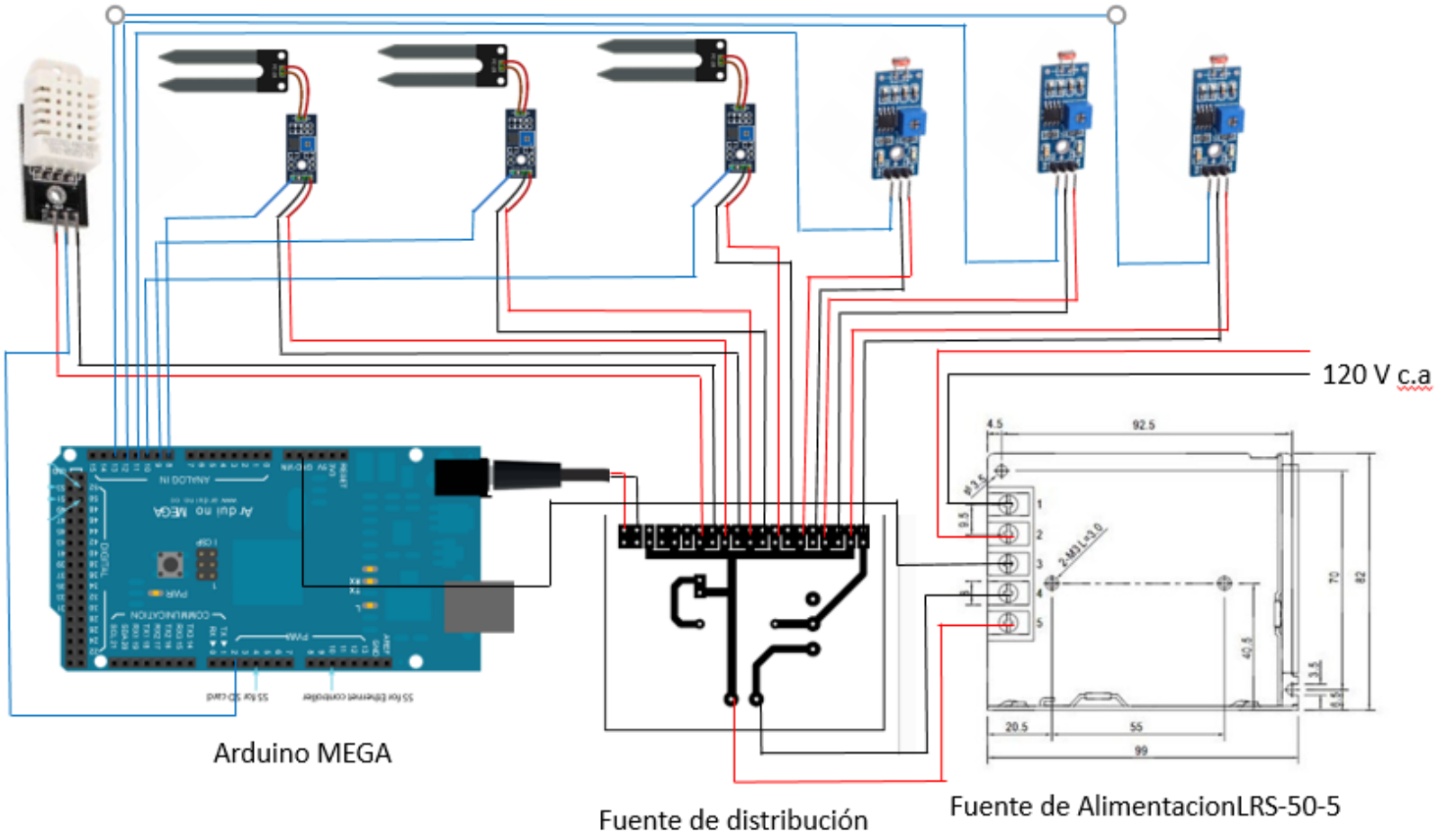


Imagen 81 Componentes soldados.

Imagen 82 Fuente terminada.

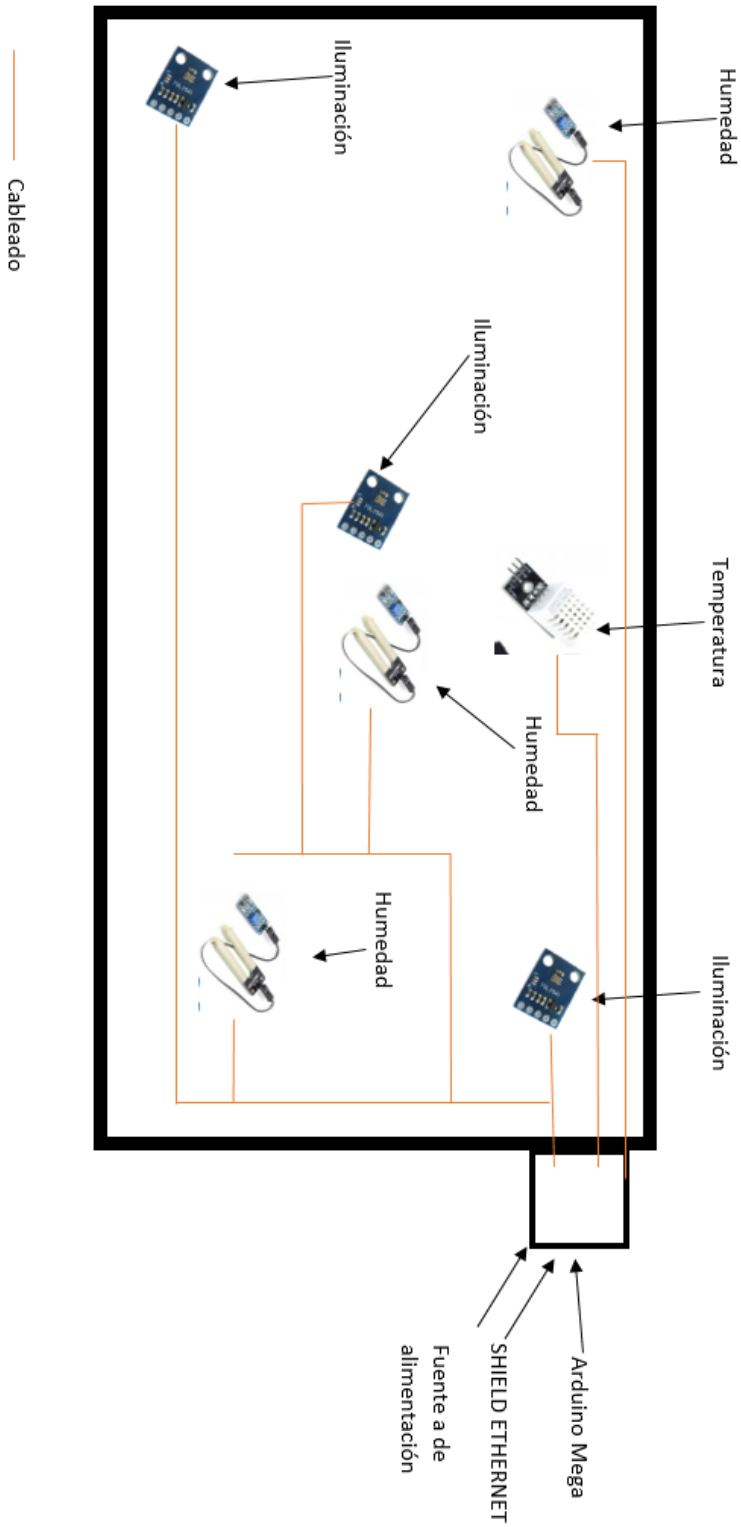
Diagrama de conexiones.




120 V c.a
Arduino MEGA
Fuente de distribución
Fuente de AlimentacionLRS-50-5

69

Distribución de los sensores en el jardín vertical.



CAPITULO IV.

Resultados y conclusiones.

Resultados.

Para poder ver los resultados realizamos pruebas para lo cual se realizó las conexiones de acuerdo al diagrama de conexionado (Imagen 83).

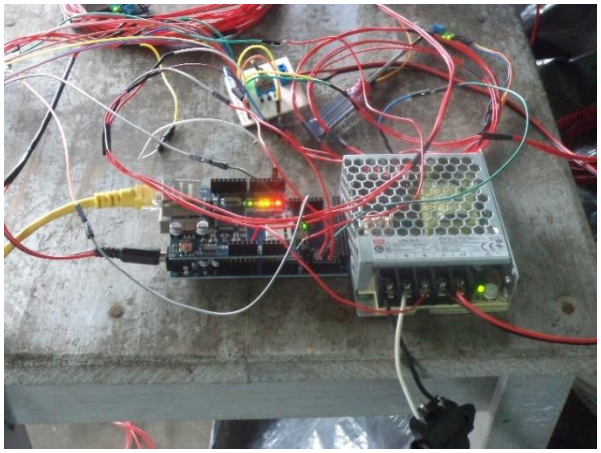


Imagen 83 Conexión de Arduino.

Entonces conectamos la fuente de distribución como el diagrama de conexionado, para así poder alimentar a todos los sensores, así como al arduino (Imagen 84).

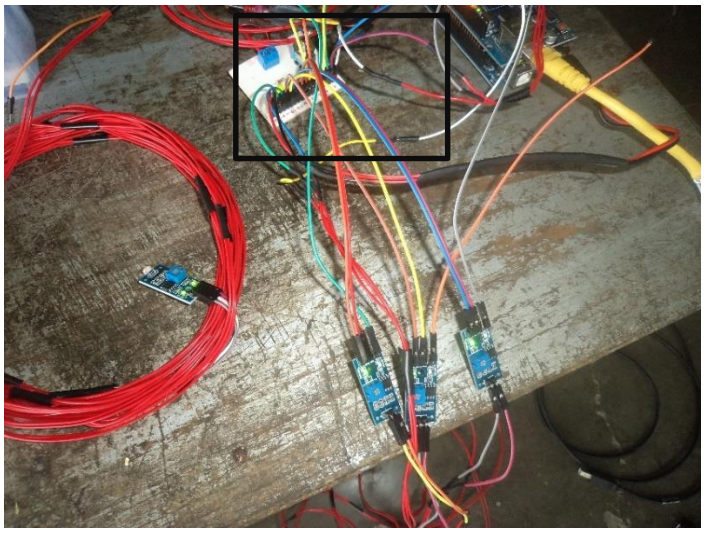


Imagen 84 Placa de distribución.

Ahora bien, para poder tener las lecturas de las sondas de humedad, es necesario introducirlas en la tierra para obtener las lecturas correctamente (Imagen 85).

Imagen 85 Sondas de humedad.

En la imagen 86 podemos ver los sensores de iluminación, el led verde indica que está en funcionamiento.

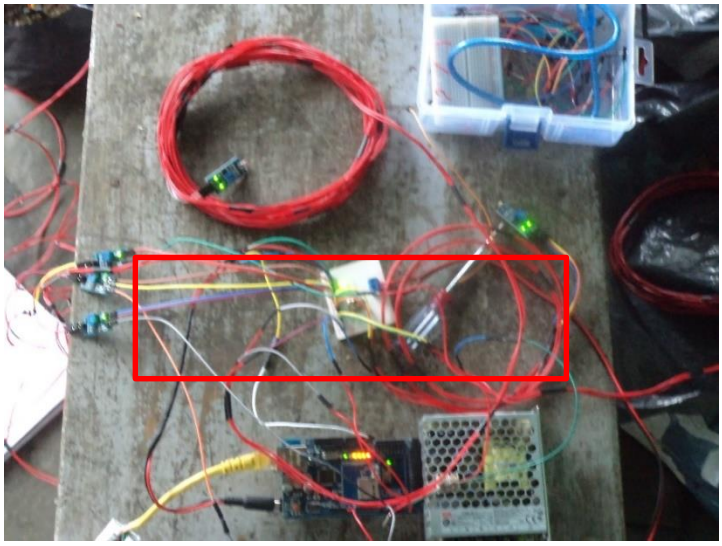


Imagen 86 Lecturas recibidas por el servidor.

Con todas las conexiones realizadas correctamente, podemos ingresar a la página del servidor para poder ver las lecturas de los sensores (Imagen 87).

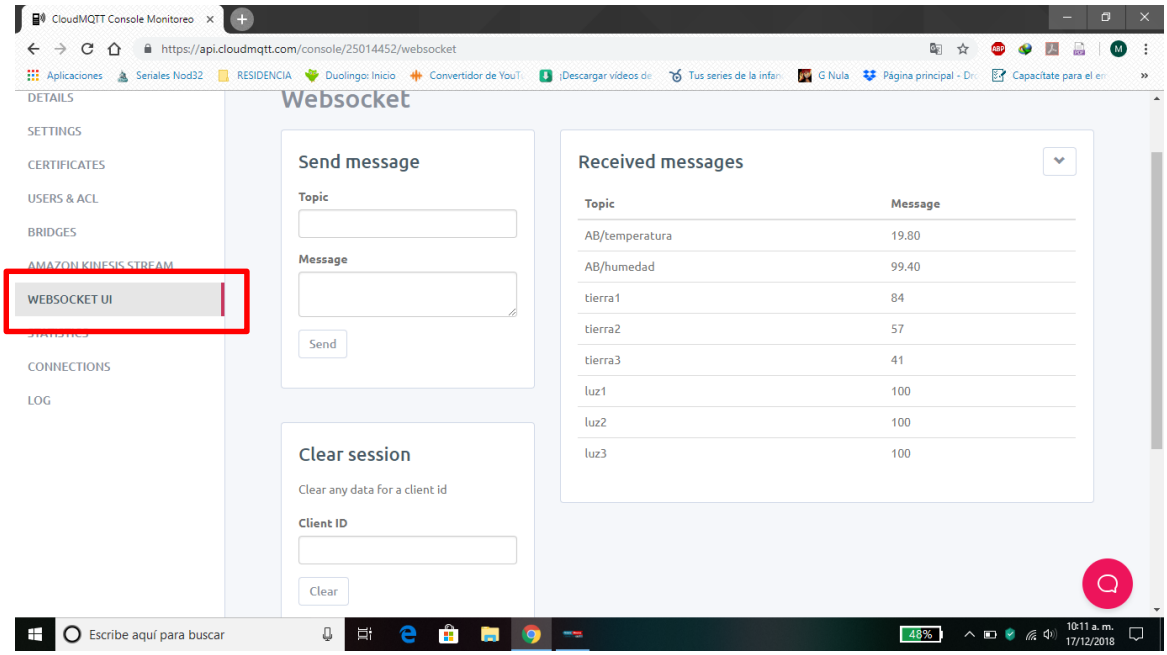


Imagen 87 Ventana de lecturas recibidas por el servidor.

Ahora verificaremos desde la App MQTT que podamos ver las lecturas que recibe el servidor, al ingresar vemos el nombre que le hemos definido “Monitoreo” y con el punto rojo es un indicador que el servidor está en línea (Imagen 88).

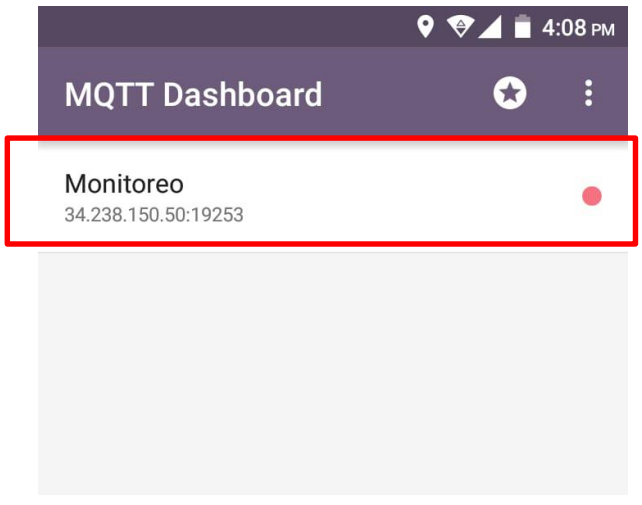


Imagen 88 Ventana principal App MQTT.

Con el servidor en línea ingresamos a “Monitoreo” para visualizar las lecturas de los parámetros (Imagen 89).

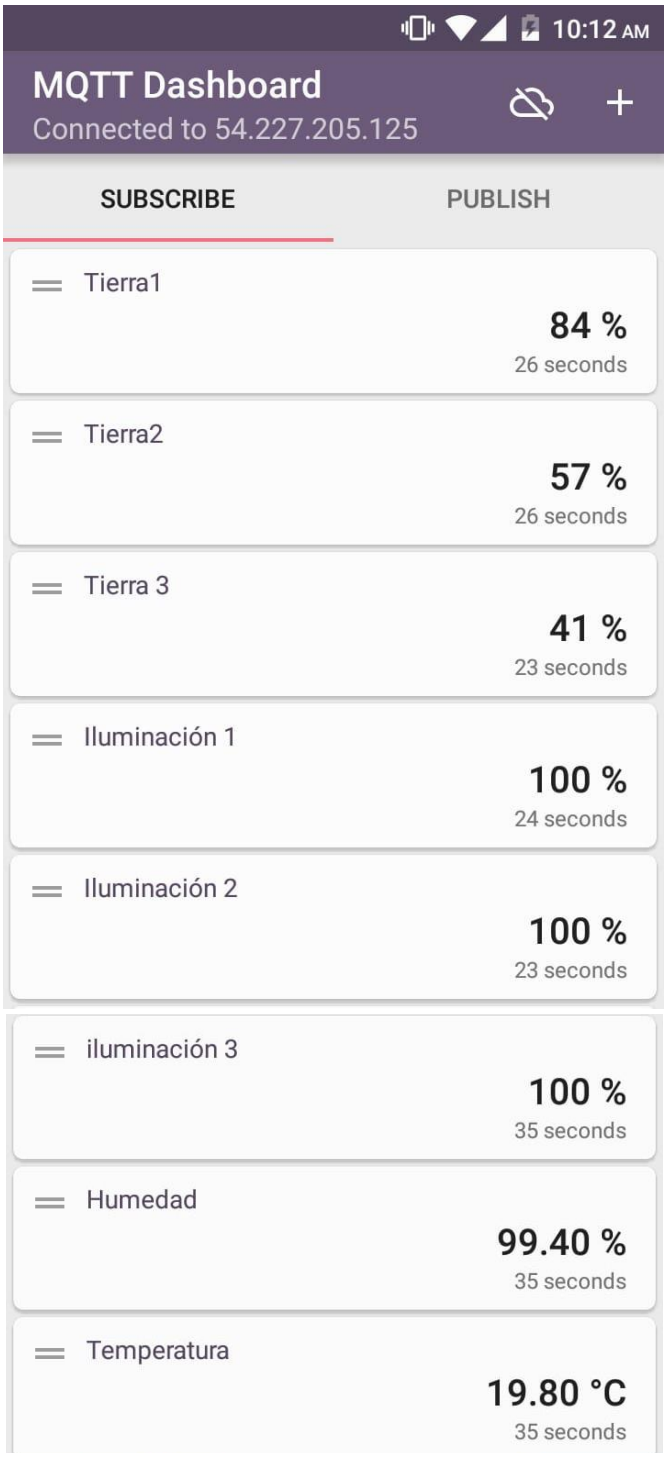


Imagen 89 Vista de las lecturas.

Ahora bien, para obtener más detalles de las lecturas, podemos ingresar a cada parámetro para ver las lecturas que en este caso se actualizan cada 5 min.

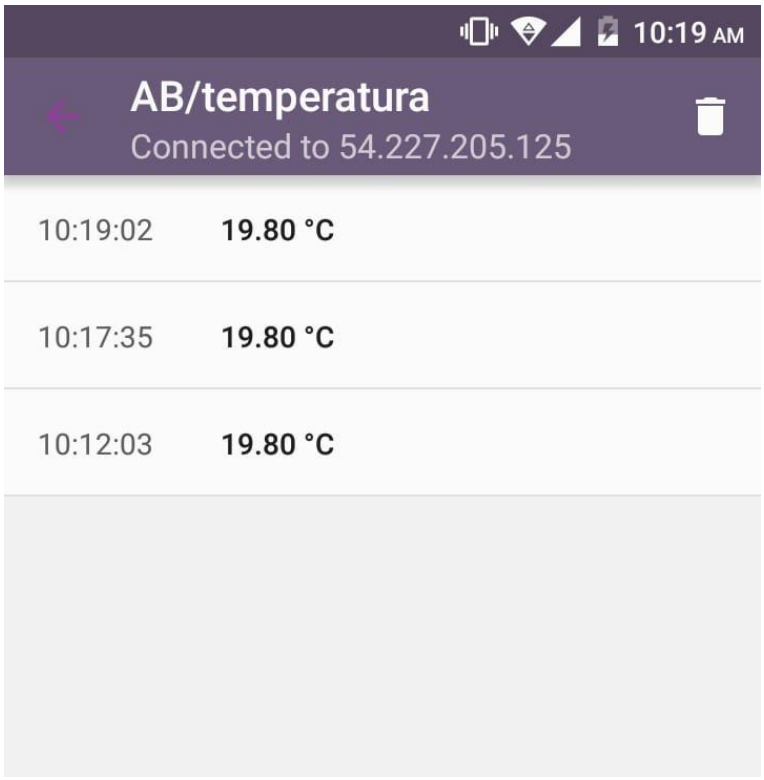


Imagen 90 Lecturas de temperatura.

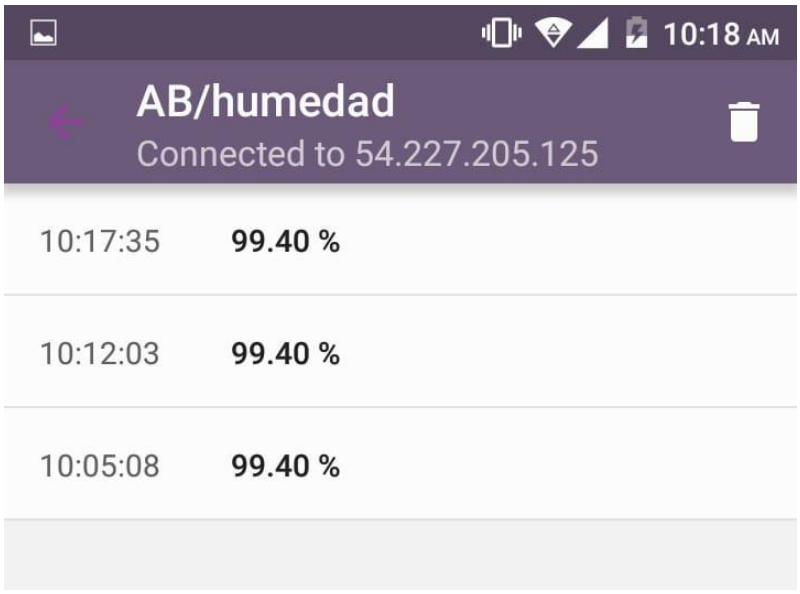


Imagen 91 Lecturas de humedad.

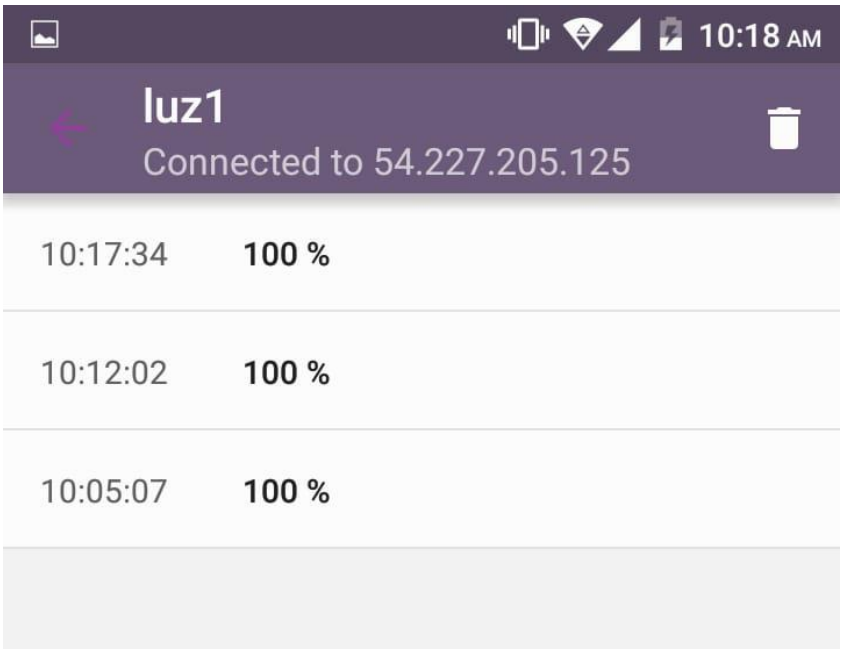


Imagen 92 Lecturas de iluminación del sensor LDR 1.

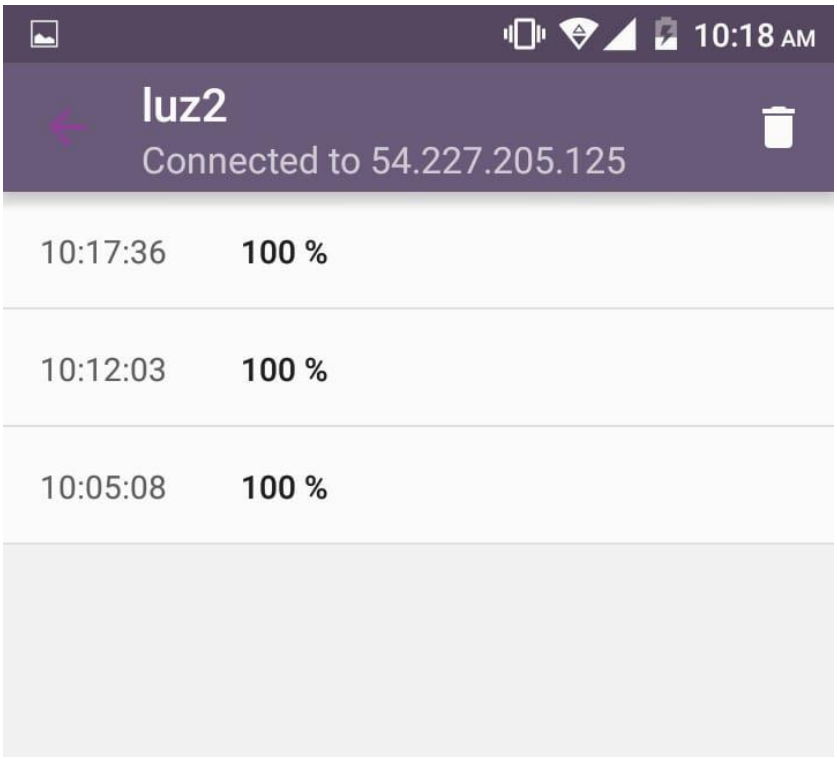


Imagen 93 Lecturas de iluminación del sensor LDR 2.

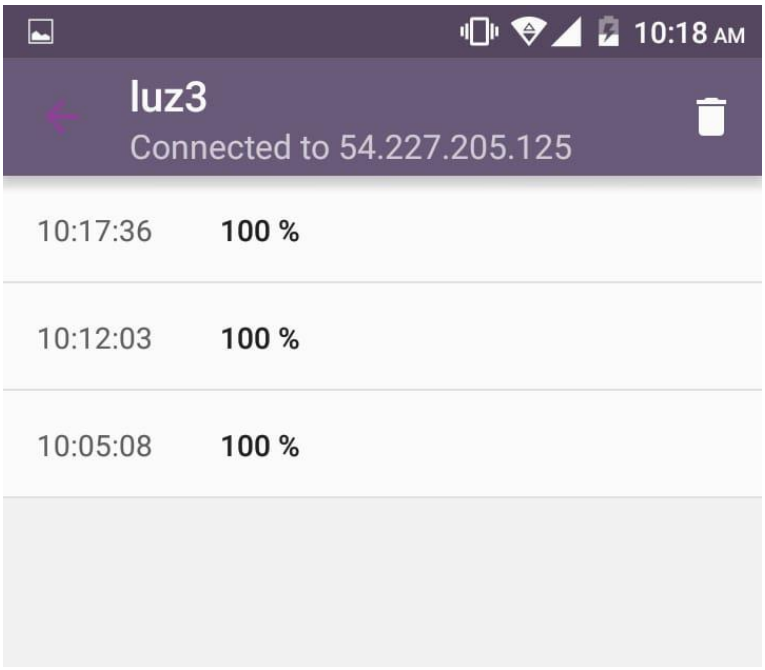


Imagen 94 Lecturas de iluminación del sensor LDR 3.

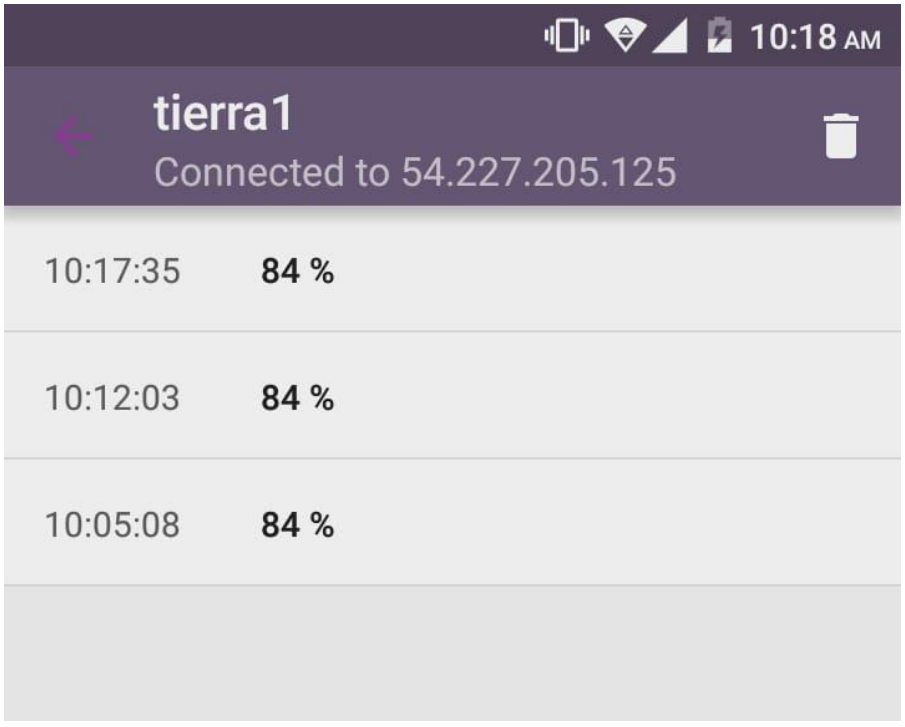


Imagen 95 Lecturas de humedad del suelo de la sonda 1.

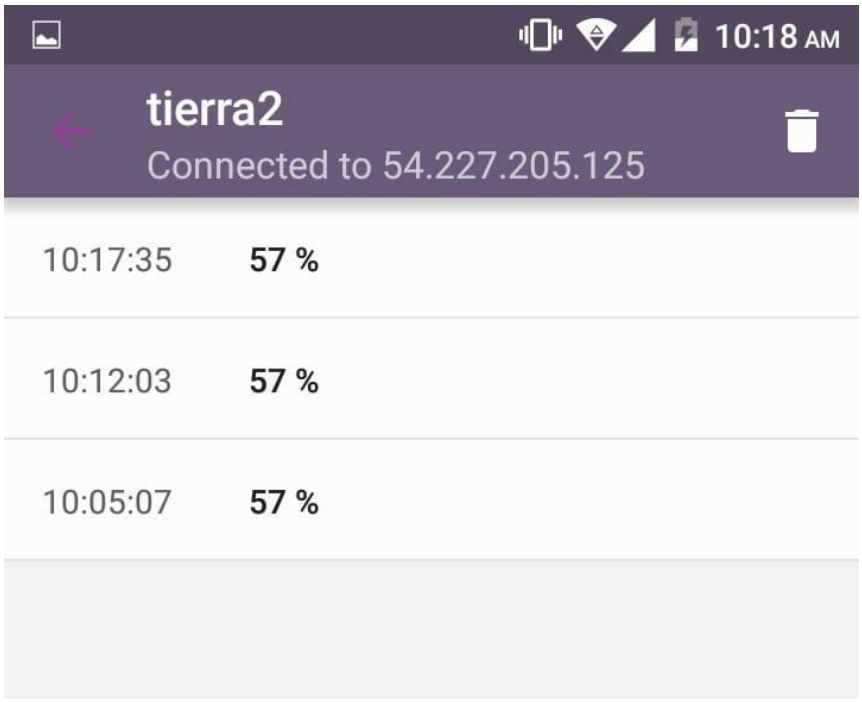


Imagen 96 Lecturas de humedad del suelo de la sonda 2.

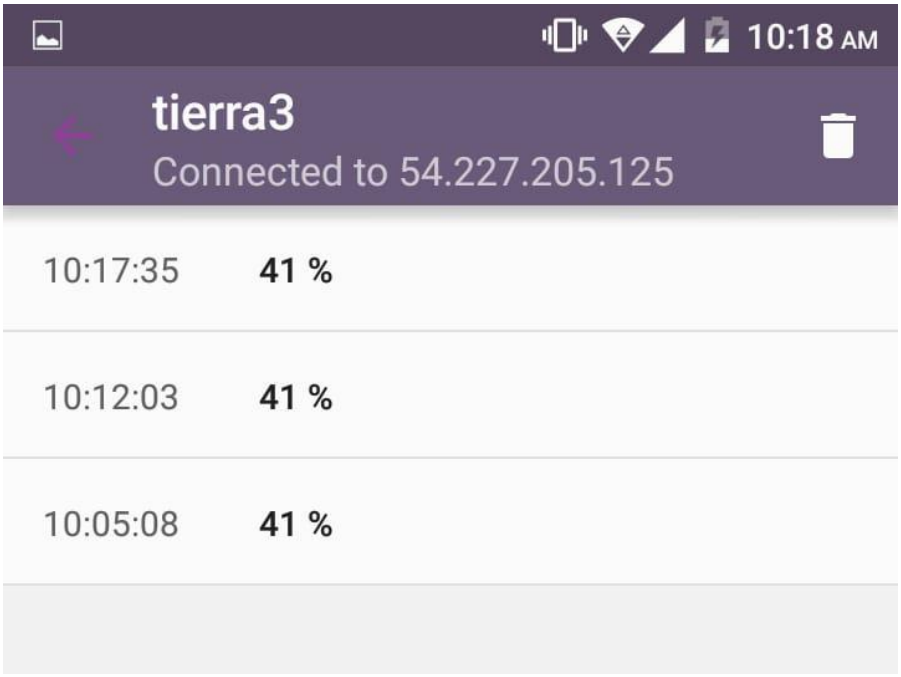


Imagen 97 Lecturas de humedad del suelo de la sonda 3.

Conclusiones.

Tras el desarrollo del proyecto se han mostrado algunas de las funciones que se puede realizar con Arduino. De este modo, se ha proporcionado toda la información relevante al funcionamiento, así como de la construcción de lo que es el sistema para monitoreo y control, para poder apreciar de lo que el sistema es capaz y de lo que no es capaz de realizar.

En conclusión, al finalizar el presente proyecto, un sistema para monitoreo y control de cultivos verticales, programable con una serie de instrucciones controladas por medio al microcontrolador Arduino.

Una de las tareas fundamentales ha sido desarrollar el programa cargado sobre la placa Arduino, mediante por el código cargador permite al microcontrolador realizar acciones en base a los parámetros que se le indiquen, realizando las lecturas según sea el orden y tiempo que se le indique. Especialmente resaltar la conexión al servidor MQTT externo que lleva a cabo el alojamiento de las lecturas obtenidas gracias a la Ethernet Shield, Así como la visualización de las lecturas desde la App MQTT para Android.

Para el correcto funcionamiento de Arduino ha sido necesario implementar el hardware necesario, para ello se ha detallado los circuitos, así como los diagramas de conexionado a la hora de desarrollar el sistema.

Una de las mayores dificultades que se presentaron al desarrollo del sistema, fue realizar la trasferencia de los datos obtenidos de los sensores al servidor, para lo cual fue necesario realizar las pruebas en diferentes servidores para probar con cuál de ellos será el más apropiado la alojar los datos; por lo cual se optó por el servidor MQTT.

Bibliografía.

Al-Kodmany,K. (2018). The Vertical Farm: A Review of Developments and Implications for the Vertical City. Buildings 2018, 8, 24.

Besthorn, F.H. (2013). Vertical farming: Social work and sustainable urban agriculture in an age of global food crises. Aust. Soc. Work 2013, 66, 187–203

Brown, T. (2008). Design Thinking. Harvard Business Review. June. U.S.A

Carruthers, S. (2007). Challenges faced by the Hydroponics Industry Worldwide. Acta Hortic. 742, 23-27

Despommier, D. (2010). The Vertical Farm: Feeding the World in the 21st Century; Thomas Dunne Books: New York, NY, USA.

Despommier, D. (2014). Encyclopedia of Food and Agricultural Ethics (Vertical Farms in Horticulture); Springer: Dordrecht, The Netherlands.

Sosa Cruz, R., (2017). ¿Qué es la industria 4?0? Consultado el 11 de septiembre de 2018. http://www.conacytprensa.mx/index.php/sociedad/politica-cientifica/18282-la-industria-4-0