# UNIVERSIDAD NACIONAL DE INGENIERÍA

FACULTAD DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA

**TESIS**

# DISEÑO E IMPLEMENTACIÓN DE UNA ARQUITECTURA IOT DE BAJO COSTO APLICADA AL CONTROL REMOTO DE HOGARES

PARA OBTENER EL TÍTULO PROFESIONAL DE:

**INGENIERO ELECTRÓNICO**

**ELABORADO POR:**

VÍCTOR ROLANDO TOMANGUILLA COLLAZOS

**ASESOR**

ING. ARMANDO ALBERTO CAJAHUARINGA CAMACO

**LIMA – PERÚ**

**2019**

# DISEÑO E IMPLEMENTACIÓN DE UNA ARQUITECTURA IOT DE BAJO COSTO APLICADA AL CONTROL REMOTO DE HOGARES

**Dedicatoria:**

Este trabajo se los dedico a mis padres y hermana con mucho cariño por ser la principal razón para seguir adelante.

**Agradecimiento:**

A mis padres Víctor y Elsa por el constante apoyo, motivación y por siempre impulsarme hacia el cumplimiento de todas mis metas. Agradezco también a mi hermana Katherine por su cariño incondicional y por siempre brindarme soporte en los momentos más difíciles.

## RESUMEN

En el presente trabajo se describe la realización de un sistema informático que permite controlar ciertas funcionalidades de algunos dispositivos dentro de una vivienda, empleando internet y dispositivos terminales, tales como celulares y computadores personales, desde locaciones remotas a dicha vivienda.

El diseño del sistema se ha realizado teniendo en cuenta ciertos criterios, tales como el costo y la complejidad en la implementación. Para dar cumplimiento a dichos criterios se ha seleccionado adecuadamente el hardware, el cual permita una solución de bajo costo accesible para nuestra realidad; así como también se ha estudiado cuidadosamente la complejidad de los conocimientos empleados para el desarrollo del mismo, de tal manera de que éste sea sencillo de implementar.

Se han realizado múltiples ensayos para probar el desempeño del sistema en un ambiente real, es decir en una vivienda habitada por potenciales usuarios, de tal manera que se pueda identificar posibles errores y poder corregirlos oportunamente. El propósito de las mismas es lograr un sistema lo más estable posible.

## ABSTRACT

In the present work we describe the realization of an informatics system that allows controlling certain functionalities of some devices in a home, using an internet connection and terminal devices, such as cell phones or personal computers, this control can be done from remote locations of that home.

The design of the system has been done considering certain criteria such as cost and implementation's complexity. To comply with these criteria, we have adequately selected the hardware, which allows a low-cost solution that could be accessed in our reality; as well as we have carefully studied the complexity of the concepts used for the system's development, in such a way that this solution is easy to implement.

We have done lots of trials to test the system performance in a real environment, that is, we have tested the system in an inhabited home with potential users, in such a way that we can identify possible errors and can fix them in a timely manner. The purpose of these trials is to achieve a system as stable as possible.

**ÍNDICE**

**CAPÍTULO III**



**CAPÍTULO IV**

## ÍNDICE DE FIGURAS

## ÍNDICE DE TABLAS

## GLOSARIO DE TÉRMINOS

| | |
|---|---|
| A | Amperios |
| AC | Alternating Current |
| ADC | Analog-to-Digital Converter |
| CoAP | Constrained Application Protocol |
| CSI | Camera Serial Interface |
| CSS | Cascading Style Sheets |
| DC | Direct Current |
| DNS | Domain Name System |
| DSI | Display Serial Interface |
| EUR | Euros |
| g | Gramos |
| GB | Gygabyte |
| GPIO | General Purpose Input/Output |
| GUI | Graphical User Interface |
| HDMI | High-Definition Multimedia Interface |
| HTML | HyperText Markup Language |
| HTTP | Hypertext Transfer Protocol |
| HTTPS | Hypertext Transfer Protocol Secure |
| IDE | Integrated Development Environment |
| IEEE | Institute of Electrical and Electronics Engineers |
| I/O | Input/Output |
| IoT | Internet de las cosas |
| IP | Internet Protocol |
| ISO | International Organization for Standardization |
| KB | Kilobyte |
| KNX | Abreviación de Konnex |
| LON | Local Operating Network |
| LTS | Long Term Support |
| Mbits/s | Megabits por segundo |
| MHz | Mega Hertz |
| mm | Milímetros |
| MQTT | Message Queue Telemetry Transport |
| NOOBS | New Out Of Box Software |
| PCB | Printed Circuit Board |

| | |
|---|---|
| PHP | Personal Hypertext Processor |
| RAM | Random Access Memory |
| SD | Secure Digital |
| SSH | Secure Shell |
| TV | Televisor |
| µP | Microprocesador |
| URL | Uniform Resource Locator |
| USB | Universal Serial Bus |
| V | Voltios |

## PRÓLOGO

El internet, desde su creación ha revolucionado el mundo de múltiples maneras, ya sea facilitándonos el acceso a la información o acortando distancias entre las personas. En los primeros días de su creación, internet únicamente era usado por un grupo selecto de personas, pero con el pasar del tiempo se fue convirtiendo en una tecnología a la que cada vez más personas tenían acceso. En nuestros días, es muy difícil concebir la idea de que un individuo no esté conectado a internet. Basándonos en este hecho, nace la posibilidad de desarrollar un sistema de automatización de tareas del hogar, que pueda ser manipulado desde cualquier parte del mundo, por un usuario con acceso a internet. La idea detrás de este sistema es modernizar los hogares, brindando comodidad a los mismos. Incluso podría servir de ayuda para personas con limitaciones físicas, tales como ancianos, niños menores o personas con discapacidad, cambiando su forma de interactuar con el hogar y mejorando su calidad de vida.

Por lo expuesto anteriormente, el proyecto de tesis tiene como objetivo la implementación de un sistema domótico que permita controlar el funcionamiento de ciertos aparatos del hogar desde cualquier parte del mundo mediante dispositivos conectados a internet. Para esta implementación se consideró un hogar con tres ambientes. En el primer ambiente, este sistema permite controlar el funcionamiento de una cocina a gas, un hervidor eléctrico y las luces del ambiente. Para el caso de los otros dos ambientes, el sistema únicamente permite el control de encendido y apagado de las luces de los mismos. La única limitación de este diseño es que el usuario no esté conectado a internet y por ende, no pueda realizar el control de los dispositivos mencionados.

El sistema anteriormente descrito pretende ser de bajo costo, lo más sencillo posible de implementar, compatible con cualquier dispositivo que tenga una conexión a internet y con componentes que estén disponibles en nuestro mercado local. Además, para su implementación se usarán tecnologías modernas como Raspberry y Arduino; así como también, servicios en la nube. Al ser esta plataforma ejecutada en la nube, se asegura una implementación estable, ya que los proveedores de servidores en la nube garantizan la operatividad de los mismos todo el tiempo.

Para el desarrollo de este proyecto de tesis fue necesario la recopilación de información de múltiples fuentes, siendo la principal de éstas, las obtenidas de páginas web en internet. Se pudieron comprobar ciertos conceptos teóricos y el código de los programas, esto gracias a la naturaleza práctica del proyecto. Adicionalmente, se validaron las hipótesis del trabajo mediante una encuesta realizada a un grupo de personas que probaron la plataforma.

# CAPÍTULO I

# INTRODUCCIÓN

## 1.1. Generalidades

El Internet de las cosas (IoT por sus siglas en inglés) es un concepto que ha tomado apreciable notoriedad en las últimas décadas, sobre todo, de la aplicación de este mismo en la tecnología de la domótica. La idea detrás de este concepto es proveer herramientas tecnológicas para crear espacios más cómodos y gestionar la seguridad en las viviendas.

En Perú el tema de la domótica es relativamente nuevo y en ciertos ámbitos desconocido. La principal razón es que este tipo de tecnología no es muy difundida en nuestro territorio. El número de construcciones que cuentan con este tipo de tecnología dentro del país es limitado y por lo general están relacionadas al ámbito empresarial, esto debido a que suelen ser muy costosas de implementar [1].

En nuestro país existen una serie de edificaciones que pueden ser consideradas como inteligentes, dentro de ellas están la nueva sede del Banco de la Nación [2], el nuevo Centro de Convenciones de Lima [3], la nueva sede de la Biblioteca Nacional [4], el campus de la Universidad de Ingeniería y Tecnología (UTEC) [5], entre otros.

Existen empresas que brindan soluciones de Domótica para hogares en Perú ([6], [7] y [8]), muchas de las cuales son muy costosas, complejas y las cuales requieren programas de mantenimiento especializados para sus equipos que también generan costos adicionales al usuario. Las empresas que ofrecen este tipo de productos y servicios tampoco son muy populares en nuestro país.

También se han realizado trabajos de investigación (tesis) relacionados al área de domótica ([9] y [10]), los cuales implementan soluciones domóticas de bajo costo. Las principales desventajas de estas propuestas son que únicamente permiten el control del hogar de manera local (es decir dentro de la vivienda) y que este control solamente se realice

mediante dispositivos con un sistema operativo en particular, reduciendo así su rango de aplicabilidad.

## 1.2. Problemática

Como se expuso en la sección anterior, en nuestro país la domótica es un aspecto que no es muy popular en estos días. Muy pocas personas conocen acerca de él y las que conocen del tema muchas veces temen dar el gran paso.

Una razón de esta escasa popularidad es la poca difusión de esta tecnología entre nuestros compatriotas. Conociendo las múltiples ventajas con las que cuenta la domótica, las personas considerarán hacerla parte de sus vidas.

Otro aspecto que no hace tan atractivo a este tipo de sistemas es el costo que involucra su implementación en los hogares, el cual muchas veces suele ser alto. Buscando en internet acerca de presupuestos sobre domótica, se pueden encontrar muchas opciones. Por ejemplo, la empresa española Domintell nos ofrece un presupuesto online [11], que dependiendo de la aplicación podría llegar a costar hasta 18700 EUR (sin contar el precio por soporte y mantenimiento), costo que para personas comunes es demasiado elevado. Si bien es cierto estos costos son válidos en España y quizá no en Perú, pero nos sirve para tener una idea de cuan costoso puede llegar a ser un sistema de estos.

El último motivo del poco uso de la domótica, es que ésta, para su control e interacción con los dispositivos en el hogar, usa interfaces complicadas y que por lo general no son tan familiares al usuario común.

Ante esta realidad, se hace necesario la creación de una plataforma de domótica de bajo costo, que sea accesible a la mayor cantidad de usuarios posibles. También debe estar por compuesta por equipos comunes con interfaces que sean amigables para todos los usuarios. Adicionalmente, esta interfaz debe ser lo más sencilla posible de implementar y dar mantenimiento, esta última tarea incluso debería poder ser hecha por el usuario.

## 1.3. Objetivos

Los objetivos de este trabajo se dividen en dos tipos los cuales se detallan a continuación.

### 1.3.1. Objetivo General

El objetivo de este trabajo es desarrollar un sistema de bajo costo que permita controlar remotamente el funcionamiento de algunos artefactos del hogar mediante el uso de interfaces móvil y web, a través de protocolos de internet.

### 1.3.2. Objetivos Específicos

- Diseñar la arquitectura de red que sirva como soporte del sistema.
- Desarrollar los protocolos para la comunicación entre los diferentes dispositivos que componen el sistema.
- Seleccionar los actuadores adecuados para el control de luces, una cocina a gas y un hervidor eléctrico dentro de la vivienda.
- Configurar los servidores en la nube, necesarios para brindar el soporte del control remoto.
- Programar las interfaces móvil y web para la interacción con el usuario.

## 1.4. Hipótesis

Al igual que los objetivos, las hipótesis de este trabajo también se dividen en dos tipos los cuales se detallan a continuación.

### 1.4.1. Hipótesis General

Es posible implementar un sistema domótico basado en IoT de bajo costo que permita controlar funciones básicas en los hogares, desde locaciones remotas al mismo usando interfaces amigables al usuario.

### 1.4.2. Hipótesis Específicas

- Es posible diseñar una arquitectura de red adecuada para el correcto intercambio de información entre los dispositivos que componen el sistema domótico.
- Es posible desarrollar algoritmos adecuados e implementarlos exitosamente para que funcionen como protocolos de comunicación e interacción entre los diversos dispositivos que componen el sistema.
- Es posible configurar adecuadamente los servidores para que sirvan de soporte a las interfaces móvil y web para el usuario.

# CAPÍTULO II

# FUNDAMENTO TEÓRICO

## 2.1. Introducción

En el presente capítulo se detallan todos los conceptos teóricos involucrados en la implementación de la plataforma de control remoto de hogares. Los conceptos van desde lo más general hasta lo más particular. Se cubren todo el detalle, tanto del hardware, así como también del software empleados para la implementación de la plataforma.

## 2.2. Internet de las Cosas

El Internet de las Cosas (IoT) se percibe como una extensión del concepto de internet convencional, en donde todas las cosas cotidianas se conectan a internet para establecer comunicación entre ellas (ver Figura 2.1), así como definir maneras para poder controlarlas [12]. No hay ninguna limitación en cuanto a los objetos que se pueden conectar a internet, siendo los más comunes en nuestros días los Smart TV, cámaras, accesorios “wearables” [13], entre otros.


**Figura 2.1** Imagen ilustrativa del concepto de IoT

(Fuente: SofOS [14])

Para lograr integrar dichos objetos en internet, se hacen uso de sistemas embebidos (como por ejemplo Arduino), los cuales proveen interfaces de hardware que facilitan la tarea de integración. [15]

En la actualidad este tipo de tecnología ya cuenta con un amplio soporte, es decir ya existen protocolos y normas que la estandarizan. La principal institución que brinda los estándares para esta tecnología es el Instituto de Ingeniería Eléctrica y Electrónica (IEEE por sus siglas en inglés) [12]. Para el caso de los protocolos, dentro de los más comunes se encuentran el CoAP [16] y el MQTT [17].

Si bien es cierto esta tecnología tiene muchas ventajas y en estos días se encuentra ampliamente desarrollada, aún enfrenta múltiples retos, siendo el principal de estos es la cuestión de ciberseguridad. Toda la data generada por los objetos conectados a internet, puede ser usada por los cibercriminales para fines maliciosos. [12]

## 2.3. Domótica

La domótica es la tecnología que permite la conexión de los dispositivos del hogar mediante una red interna con la finalidad de que los mismos puedan ser controlados a distancia. Es decir, la domótica permite la automatización de los hogares. Esta tecnología permite una gestión eficiente del uso de la energía, así como también aporta seguridad y confort a las viviendas (como se aprecia en la Figura 2.2). [18]

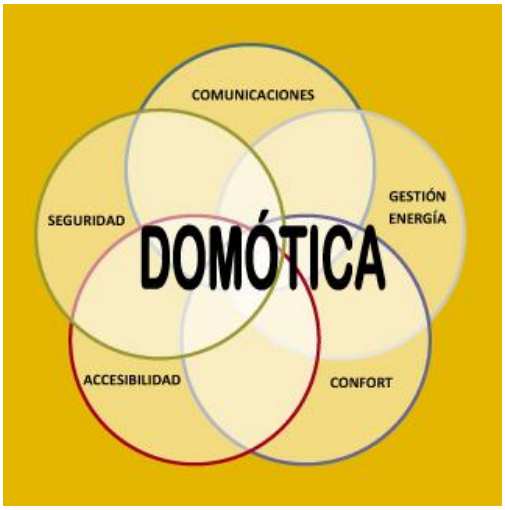



**Figura 2.2** Imagen ilustrativa del concepto de domótica
(Fuente: CEDOM [18])

Para lograr este objetivo, es necesario el desarrollo de sistemas que involucren tanto el diseño de hardware como el de software, y cuyo producto final permita la gestión de dispositivos del hogar.

En la actualidad existen dos estándares internacionales de domótica: el KNX [19] de Konnex Associations y el LonWorks [20] de LonMark Association, los cuáles brindan normas y especificaciones para el diseño eficiente de sistemas domóticos, así como garantizar la compatibilidad entre los mismos.

## 2.4. Sistemas Embebidos

Los sistemas Embebidos son sistemas electrónicos que son diseñados para cumplir un rango de funcionalidades, las cuales, por lo general, no pueden ser cambiadas o programadas por el usuario final de la misma forma como se programa en un computador personal. En otras palabras, un sistema embebido está diseñado para cumplir necesidades específicas. [21]

Algunos ejemplos de este tipo de sistemas son las microcomputadoras que controlan la lógica de algunos electrodomésticos, tales como televisores, lavadoras, radios, etc. Incluso las computadoras personales cuentan con sistemas embebidos para el control de ciertas funcionalidades y dispositivos periféricos. También es bien común encontrar sistemas embebidos en automóviles para el control, por ejemplo, de los sistemas de frenos, bloqueo, transmisión, etc. Así como estos, hay otros muchos ejemplos más de la aplicación de sistemas embebidos en objetos del día a día, y de los cuáles no somos conscientes.

Los sistemas embebidos usualmente se encuentran compuestos por las siguientes partes [21]:

- **Procesador:** Como su mismo nombre lo dice, es el sistema encargado de las labores de procesamiento de la data con la que interactúa el sistema embebido.
- **Memoria:** Provee el almacenamiento de la información necesaria para el correcto funcionamiento del sistema.
- **Periféricos:** Son todos los dispositivos que permiten la comunicación del sistema embebido con el exterior.
- **Software:** Es la parte del sistema que brinda la funcionalidad al mismo, es decir le provee las herramientas para su inicialización, configuración, y mantenimiento.
- **Algoritmos:** Son los componentes que describen el comportamiento de nuestro sistema embebido.

Hoy en día existen muchas tecnologías que facilitan el desarrollo de sistemas embebidos. Por fines de conveniencia del presente trabajo, sólo se abordarán dos de estas tecnologías: Arduino y Raspberry.

### 2.4.1. Arduino

Arduino es un tarjeta electrónica open-source [22] de bajo costo, que es usada para el desarrollo de prototipos. Esta plataforma permite la manipulación de entradas y salidas a través de sus interfaces. La tarjeta Arduino es sencilla de usar, por ello desde su creación, ha sido involucrada en múltiples proyectos, tanto a nivel empírico como científico. Para su

programación se usa el lenguaje de programación Arduino (muy parecido al lenguaje de programación C), dentro de un entorno de desarrollo integrado conocido como el Arduino IDE. [23]

Dentro de las ventajas de usar Arduino sobre otras plataformas de desarrollo se encuentran [24]:

- **Bajo costo:** Las placas Arduino son las más asequibles comparadas con otras. La versión más básica de esta tarjeta puede llegar a costar en Perú unos S/. 18,00.
- **Multiplataforma:** El Arduino IDE puede funcionar en los sistemas operativos Windows, Linux y Macintosh OS X.
- **Entorno de Programación Simple y Claro:** El Arduino IDE es muy fácil de usar incluso para personas ajenas al mundo de la programación.
- **Hardware y Software Libre:** Todas las especificaciones de Arduino, tanto en hardware como en software, se encuentran abiertas a todo el mundo bajo licencia *Creative Commons*.

Existen múltiples versiones de la tarjeta en el mercado actual, cada una provista de recursos de hardware para determinada aplicación. Dentro de las modelos más populares se encuentran el NANO, UNO y MEGA. Debido a que en el presente proyecto se hará uso de la tarjeta Arduino NANO, únicamente se detallarán los componentes de hardware de la misma, la Tabla N° 2.1 muestra las características más resaltantes de dicha tarjeta.

Para el caso de Arduino NANO, cada uno de los pines de su microcontrolador se encuentra asignado para una determinada función dentro de la tarjeta y dicha función no puede ser modificada por el usuario. La Figura 2.3 muestra la distribución de los pines de la tarjeta Arduino NANO. En la Tabla N° 2.2 se describe la función de cada uno de esos pines.

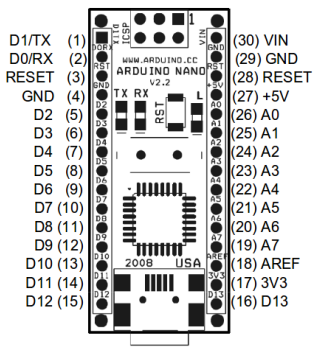



**Figura 2.3** Distribución de pines del Arduino NANO

(Fuente: Arduino [26])

**Tabla N° 2.1** Especificaciones de la placa Arduino NANO

| | |
|---|---|
| Microcontrolador | ATmega328 (AVR) |
| Voltaje de Operación | 5 V |
| Memoria Flash | 32 KB (2 KB para bootloader) |
| SRAM | 2 KB |
| Reloj | 16 MHz |
| Entradas analógicas | 6 |
| Puertos I/O | 14 |
| Salidas PWM | 6 |
| Tamaño | 18 x 45 mm |
| Peso | 7 g |

(Fuente: Arduino [25])

**Tabla N° 2.2** Descripción de los pines del Arduino NANO (Fuente: Arduino [26])

| PIN N° | NOMBRE | TIPO | DESCRIPCIÓN |
|---|---|---|---|
| 1-2, 5-16 | D0 - D13 | I/O | Puertos Digitales |
| 3, 28 | RESET | Input | Reset (activo bajo) |
| 4, 29 | GND | PWR | Tierra |
| 17 | 3V3 | Output | Salida de 3.3V |
| 18 | AREF | Input | Referencia del ADC |
| 19 - 26 | A7 - A0 | Input | Entradas analógicas |
| 27 | +5V | I/O | Salida de +5V o Entrada de alimentación de +5V |
| 30 | VIN | PWR | Alimentación para la tarjeta |

(Fuente: Arduino [26])

**a) Programación de la placa Arduino**

Como ya se mencionó anteriormente, Arduino cuenta con una interfaz de software para su programación de la misma mediante un computador, esta interfaz lleva por nombre Arduino IDE. Dicho software cuenta con un editor de textos, una consola y una serie de botones relacionados a la programación de la placa Arduino. El rango de funcionalidades de este software es muy amplio y la descripción de cada uno de ellos sería muy extensa, por ende, no entraremos a detalle en ese aspecto. Para los fines del presente proyecto únicamente

se empleó el editor de textos, el botón de compilación (✓) y el botón de descarga del programa en la placa (→). La Figura 2.4 muestra esta interfaz de software.

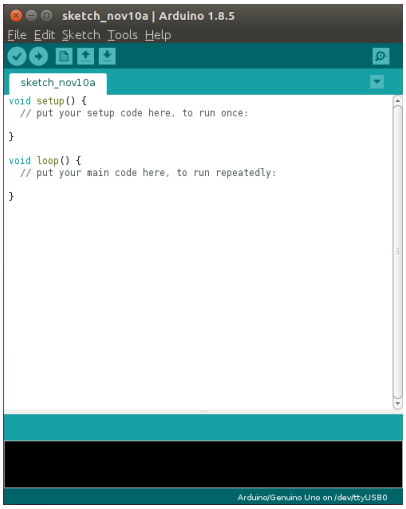



**Figura 2.4** Interfaz del Arduino IDE

(Fuente: Elaboración propia)

Al programa escritos para la placa Arduino se le conoce como *sketch*, estos archivos se guardan con extensión *ino*. Todos los sketches tienen una estructura básica compuesta por dos funciones principales [28]:

- **Función *setup()*:** En esta función se especifican todos los parámetros necesarios para la inicialización de la placa. En esta función se pueden configurar los puertos digitales para que se comporten como entradas o salidas, establecer la velocidad de la comunicación serial, asignar los puertos para la conexión con hardware externo, entre otras acciones de inicialización. Es importante mencionar que esta función se ejecuta solamente una vez.
- **Función *loop()*:** Esta función alberga toda la lógica del programa, es decir dentro de esta función va contenido el algoritmo de nuestro programa. A diferencia de la función setup(), la función *loop()* se ejecuta infinitamente siempre y cuando la tarjeta se encuentre energizada. Esta función es lo que en programación se conoce como bucle infinito.

El lenguaje de programación de Arduino maneja una serie de sentencias propias que permiten interactuar con los pines anteriormente descritos. Dentro de las sentencias más comunes se encuentran [28]:

- **pinMode(pin,modo):** Esta instrucción por lo general va dentro de la función *setup()* y sirve para configurar cualquiera de los pines digitales ya sea como entradas o como salidas. Esta sentencia recibe dos parámetros, el primero es el número de pin (0 - 13) y el segundo es el modo de funcionamiento (INPUT o OUTPUT).

- **digitalWrite(pin,valor):** Esta instrucción sirve para establecer el estado de un pin cuando este ha sido configurado como salida (OUTPUT). Esta sentencia recibe dos parámetros, el primero es el número de pin (0 - 13) y el segundo es el valor con el que queremos configurar la salida de dicho pin (HIGH o LOW).
- **digitalRead(pin):** Esta instrucción se usa para leer el estado de un pin que ha sido configurado como entrada digital. Esta sentencia recibe como único parámetro el pin del que deseamos leer su estado (HIGH o LOW).
- **analogWrite(pin,valor):** Esta instrucción se usa para enviar una señal analógica del tipo PWM (modulación por ancho de pulso) a través de un pin digital configurado como salida (OUTPUT) y que esté provisto con esta funcionalidad. El primer parámetro que recibe esta sentencia es el pin porque él se enviará la señal analógica y el segundo parámetro es el porcentaje del ancho del pulso especificado con un número entre 0 y 255, donde 0 denota 0% y 255 denota el 100%.
- **analogRead(pin):** Esta instrucción se usa para leer el valor de una señal ingresada a través de un pin analógico (A0 - A7). Cada pin analógico tiene conectado internamente un conversor análogo digital (ADC) de 10 bits, por lo que esta sentencia en lugar de mostrar voltaje nos mostrará un número entero comprendido entre 0 y 1023, donde 0 representa una lectura de 0V y 1023 representa una lectura de 5V.
- **delay(tiempo):** Esta sentencia permite congelar la ejecución del programa por un tiempo determinado (en milisegundos), el cual es especificado como parámetro de la misma.

También este lenguaje de programación cuenta con ciertas estructuras que permiten controlar el flujo de nuestro programa, dentro de las estructuras de control que maneja este lenguaje se encuentran [28]:

- **Estructura *if-else*:** Esta estructura permite discernir entre la ejecución de dos grupos de sentencias. El mecanismo que controla el flujo de este tipo de estructura es una condición booleana (es decir que puede tomar el valor de verdadero o falso). La manera de interpretar esta estructura es la siguiente: si la condición booleana es verdadera se ejecuta un determinado grupo de instrucciones, y si la condición es falsa se ejecuta otro grupo de instrucciones. La organización de esta estructura es la siguiente:

```
if (condición){
          Bloque de Sentencias 1;
}
else{
          Bloque de Sentencias 2;
}
```

Es importante mencionar que un *if* con su condición si pueden ir por sí solos, pero un *else* siempre tiene que estar precedido por un *if* con su respectiva condición.

- **Estructura *switch*:** Esta estructura es también del tipo condicional, solo que este caso permite discernir entre múltiples grupos de sentencias. En este tipo de estructura el control del flujo se hace con una variable cualquiera, y dependiendo del valor que tome esta variable se ejecutará un determinado grupo de sentencias. La organización de esta estructura es la siguiente:

```
switch (variable){
          case valor1:
                    Bloque de Sentencias 1;
                    break;
          case valor2:
                    Bloque de Sentencias 2;
                    break;
          case valor3:
                    Bloque de Sentencias 3;
                    break;
          default:
                    Bloque de Sentencias 4;
                    break;
}
```

En esta estructura ejemplo se consideró el escenario en el que la variable sólo podría tomar tres valores (*valor1*, *valor2* y *valor3*) pero es importante mencionar que se pueden agregar más casos a la condición y ejecutar otros bloques de sentencias. También esta estructura permite discernir entre una condición por defecto (*default*), que se ejecutará cuando la variable no tome ninguno de los valores anteriormente considerados.

- **Estructura *while*:** Esta estructura permite ejecutar un cierto grupo de sentencias tantas veces como una condición booleana sea verdadera. Este tipo de estructuras es también son conocidas como bucles. La organización de esta estructura es la siguiente:

```
while (condición) {
          Bloque de Sentencias 1;
}
```

En este tipo de estructuras es importante seleccionar adecuadamente la condición y la forma de cómo esta evoluciona para evitar caer en repeticiones infinitas o lo que comúnmente llamamos un bucle infinito.

- **Estructura *for*:** Esta estructura al igual que el *while* también es del tipo repetitiva, solo que en esta última la información de la repetición (o bucle) se especifica en una sola línea. La organización de esta estructura es la siguiente:

```
for(val_inicial; condición; val_control){
          Bloque de Sentencias 1;
}
```

Al igual que en el caso del bucle while, también se debe elegir adecuadamente la condición y como esta evoluciona para evitar caer en repeticiones infinitas.

**b) Comunicación serial con Arduino**

Otra de las funcionalidades que permite la tarjeta Arduino es la comunicación mediante el protocolo Serial. Esta comunicación se puede dar a través de los pines digitales definidos para esta función (D0 para RX y D1 para TX) o a través de la adaptación serial a USB que posee la placa. El protocolo de comunicación serial tiene ciertas consideraciones tanto hardware que gracias a Arduino son transparentes al usuario y sólo se limitan al uso de ciertas sentencias en su lenguaje de programación. Dentros de las sentencias más importantes para el manejo de la comunicación serial mediante Arduino se encuentran [28]:

- **Serial.available():** Esta sentencia permite conocer si existe información para leer en el bus serial. Devuelve el valor de 0 si no existe nada que leer y en el caso contrario devuelve un número entero conteniendo la cantidad de bytes disponibles para la lectura.
- **Serial.read():** Esta instrucción permite la lectura de información directamente del bus serial.
- **Serial.write():** Este comando permite enviar información al bus serial para su correspondiente recepción en otro dispositivo conectado a la misma.

Estos son los comandos más comunes en cuanto a la comunicación serial mediante Arduino, existen otros más pero simplemente son variantes a estos comandos básicos.

### 2.4.2. Raspberry

Raspberry es un miniordenador de bajo costo usado para múltiples propósitos. Este sistema cuenta con todos los componentes de un computador común (procesador, memoria, periféricos, etc.) incrustados en una única placa. Como cualquier ordenador, Raspberry Pi cuenta con su propio sistema operativo, en este caso de trata de un sistema operativo de nombre Raspbian. La manera más usual de programar este tipo de dispositivos, es a través del lenguaje de programación Python, cuyo intérprete viene cargado por defecto en el sistema operativo del Raspberry Pi [29]. Esto no quiere decir que no se pueda usar otros lenguajes de programación para la configuración de este tipo de tarjetas, también se pueden usar lenguajes como C, C++, Java, entre otros; siempre y cuando se disponga de sus respectivos compiladores para los mismos. El modelo que escogimos para este proyecto es el Raspberry Pi Modelo B v1.1, la cual se muestra en la Figura 2.5.

**Figura 2.5** Raspberry Pi 2 Modelo B

(Fuente: Raspberry [29])

Adicionalmente a las interfaces comunes con la que cualquier computador cuenta, Raspberry posee una serie de pines adicionales que permiten la conexión de este con otro tipo de dispositivos, agregando nuevas funcionalidades a la tarjeta. Estos pines se les conoce como GPIO (pines de entrada/salida de propósito general por sus siglas en inglés). No se entrarán en detalle acerca de esta característica de Raspberry ya que no se emplearon a la hora de desarrollar el proyecto.

Entrando en detalle de los componentes que posee este mini computador (Raspberry Pi Modelo B) podemos listar los siguientes [29]:

- 1 procesador quad-core ARM Cortex-A7@900MHz
- 1 GB de Memoria RAM
- 1 interfaz Ethernet
- 4 puertos USB
- 40 pines GPIO
- 1 puerto HDMI
- 1 conector de audio y video compuesto
- 1 interfaz para cámara (CSI)
- 1 interfaz para Display (DSI)
- 1 slot para Micro SD

Para interactuar con este tipo de tarjetas existen dos formas, una primera forma es conectando un teclado, un mouse y un monitor a la tarjeta. La segunda forma es a través de un ordenador mediante una conexión SSH (Secure Shell) entre el ordenador y la tarjeta, para esta última, tanto el Raspberry como el host remoto deben estar conectados en a una red para que sea posible su comunicación. El usuario es el que debe elegir el método con el que se sienta más cómodo.

En cuanto a la alimentación de este tipo de tarjetas dependerá de la cantidad de dispositivos externos tenga conectada a la misma. El tipo de entrada para la alimentación es del tipo micro USB. Para este proyecto se usó un cargador de celular de 5 Voltios a 2.5 Amperios para la alimentación del Raspberry Pi.

**a) Raspbian**

Raspbian es el sistema operativo exclusivo para las tarjetas Raspberry Pi, ya que se encuentra optimizado para el hardware de que este maneja. Es un tipo de sistema operativo embebido basado en GNU/Linux. Existen dos versiones de este tipo de este sistema operativo. La primera de estas es la versión completa (de Escritorio), la cual cuenta con un entorno gráfico con todos los elementos propios de este, es decir, cuenta con íconos, ventanas, un puntero para el mouse, etc.; esta versión por lo general es usada por usuarios principiantes debido a su sencilla interfaz. La segunda versión es la versión Lite que no cuenta con entorno gráfico y para su manejo únicamente se dispone de una consola; esta versión es empleada por usuarios avanzados los cuáles cuentan con conocimientos de Linux [30].

Para instalar este sistema operativo y usarlo en un Raspberry se cuentan con dos métodos que la comunidad de desarrolladores de Raspberry pone a disposición de los usuarios del mismo. Un método es a través del asistente de instalación de nombre NOOBS (New Out Of Box Software) que graba la imagen del sistema operativo en una tarjeta SD, por lo general este método es empleado por usuarios principiantes; no se entrará en detalle sobre este método ya que no fue usado en el presente trabajo. El segundo método es a través de la imagen ISO de este sistema operativo, para instalar usando este método se debe seguir el siguiente procedimiento [30]:

- En primer lugar, se debe descargar la imagen ISO de Raspbian desde su página web oficial [30].
- Después se debe grabar dicha imagen en una tarjeta SD, para ello se debe hacer uso del software Etcher [31].
- Finalmente, una vez terminado el proceso anterior, se debe insertar la memoria SD en el Raspberry, y conectar esta última a su fuente de alimentación.

Una vez instalado el sistema operativo lo que sigue es simplemente usarlo. Al conectar la tarjeta a la fuente de alimentación, ésta iniciará su proceso de encendido. Cuando la tarjeta culminó su proceso necesitaremos de una información de inicio de sesión para poder hacer uso de las funcionalidades de esta tarjeta. Por defecto la información de inicio de sesión es la siguiente [30]:

- **Usuario:** pi
- **Contraseña:** raspbian

Hasta este punto ya se tiene nuestro Raspberry listo para usar. La forma de interactuar con este dependerá de la versión instalada.

Si se instaló la versión completa se debe ejecutar el comando `sudo startx` para iniciar su entorno gráfico (ver Figura 2.6). Cuando el entorno gráfico está cargado simplemente interactuamos con él a través un teclado y un ratón [32].

Por su parte si se instaló la versión Lite (en modo consola, la cual se muestra en la Figura 2.7), es necesario conocer algunos comandos propios de sistemas operativos del tipo Linux para poder interactuar con el Raspberry. La Tabla N° 2.3 muestra algunos de los comandos más útiles del sistema operativo Linux.

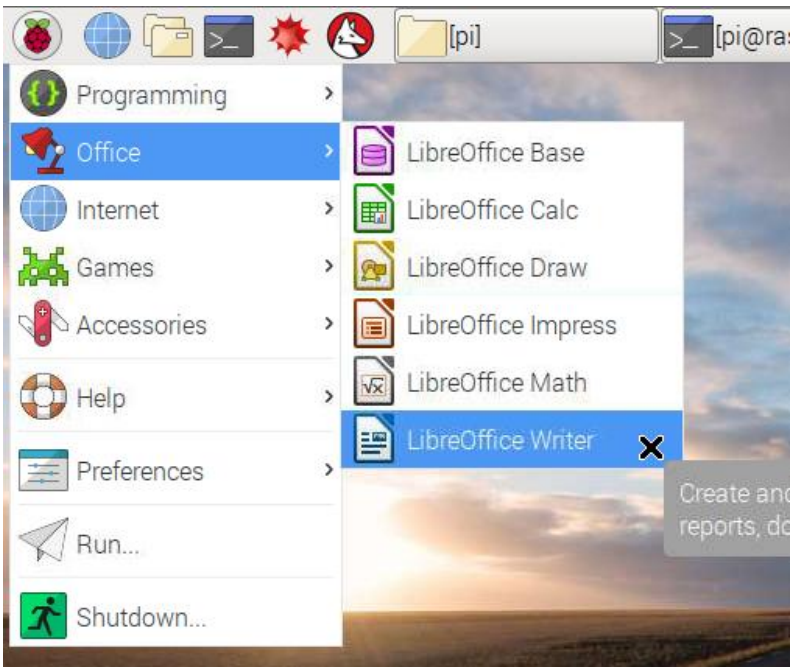



**Figura 2.6** Raspbian en modo Gráfico

(Fuente: MakeUsOf [33])

Los procesos anteriores pueden ser realizados siempre y cuando el usuario disponga de un monitor (con conector HDMI), un teclado y un ratón. En el caso de no disponer de estos dispositivos, existe otro método para acceder al Raspberry únicamente en modo consola, este es a través de una conexión SSH con el mismo mediante una conexión a una red de computadores tanto del Raspberry como del host remoto. Es importante mencionar que para este tipo de conexión el usuario debe conocer algunos comandos de Linux. Este tipo de protocolo de comunicación se describe en la siguiente sección.

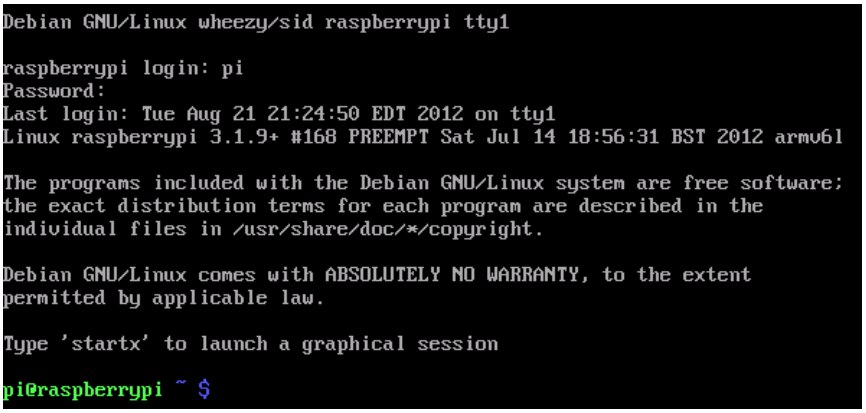



**Figura 2.7** Raspbian en modo consola

(Fuente: El Blog de Adrian [34])

**Tabla N° 2.3** Principales comandos de Linux

| COMANDO | DESCRIPCIÓN |
|---|---|
| ls | Muestra el contenido de un directorio |
| cd | Navega en un directorio |
| cd .. | Cambia al directorio anterior |
| mkdir | Crea un directorio nuevo |
| rmdir | Elimina un directorio vacío |
| cp | Copia archivos |
| mv | Mueve archivos |
| rm | Elimina archivos y directorios |
| cat | Muestra el contenido de un archivo |
| chmod | Cambia los permisos de un archivo o directorio |
| nano | Inicia el editor de textos NANO |

(Fuente: CCM [35])

**b) SSH**

SSH (acrónimo de Secure Shell) es un protocolo de comunicación entre dispositivos a través de internet. Mediante este protocolo el usuario puede conectarse a un dispositivo remoto para controlar sus funcionalidades. En este protocolo toda la información viaja encriptada para evitar que otros usuarios puedan tener acceso directo a ella. Este protocolo nace como la evolución del protocolo Telnet, el cual también es un protocolo de comunicación remoto solo que en este último las comunicaciones no son encriptadas. SSH utiliza tres técnicas para el cifrado de sus comunicaciones [36]:

- **Encriptación Simétrica:** en este tipo de encriptación se usa una única clave tanto para el cifrado como para el descifrado de los mensajes.
- **Encriptación Asimétrica:** en este caso se usan dos claves, una para el cifrado y otra para el descifrado del mensaje.
- **Hashing:** para este caso se usan funciones *hash* [37] unidireccionales para la encriptación del mensaje.

La conexión SSH es del tipo cliente-servidor a través del puerto TCP 22. Para poder realizar la conexión es necesario que el dispositivo remoto tenga instalado y configurado un servidor SSH. En el caso del dispositivo local, este debe tener instalado un cliente. En el caso de que el dispositivo local esté ejecutando un sistema operativo Windows, el cliente más conocido es PuTTY [36] (ver Figura 2.8).

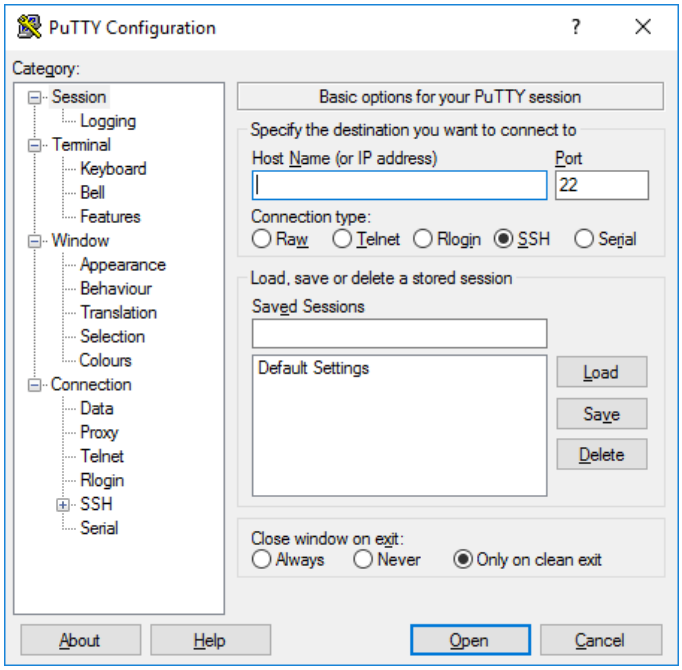



**Figura 2.8** Conexión SSH desde PuTTY

(Fuente: PuTTY [38])

Si se trata de un sistema operativo Linux (en este caso bajo la distribución Ubuntu 16.04 LTS), se debe instalar un servidor SSH, para ello se debe ejecutar en el siguiente comando en la consola de Ubuntu [39]:

```
sudo apt-get install openssh-server
```

En el paquete de software instalado usando el comando anterior viene incluido tanto el cliente como el servidor SSH. Para iniciar una comunicación remota mediante SSH, se debe ejecutar el siguiente comando en la consola del dispositivo local (ver Figura 2.9), que en nuestro caso de trata de un computador con sistema operativo Ubuntu 16.04 [36]:

```
ssh usuario_host_remoto@ip_host_remoto
```

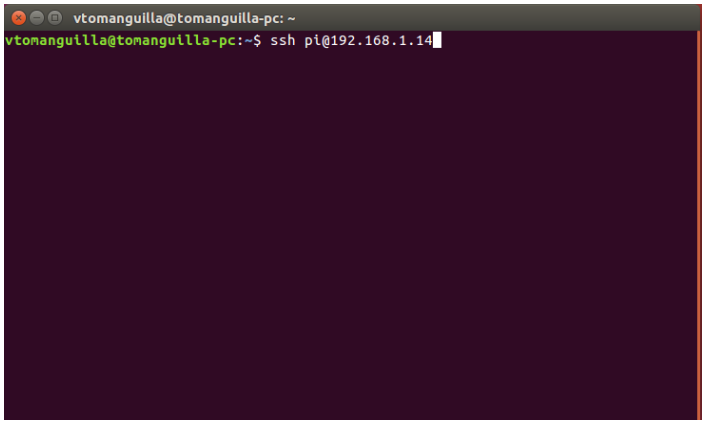



**Figura 2.9** Conexión SSH desde la Terminal de Ubuntu

(Fuente: Elaboración propia)

Para acceder al Raspberry mediante SSH, es necesario configurar este servicio en el mismo. Para ello, se debe tipear en su consola el siguiente comando [40]:

```
sudo raspi-config
```

Al ejecutar este comando nos mostrará un menú de configuración (ver Figuras 2.10 y 2.11). En dicho menú seleccionamos la opción *Advance Options*, y luego la opción *SSH* y la habilitamos.

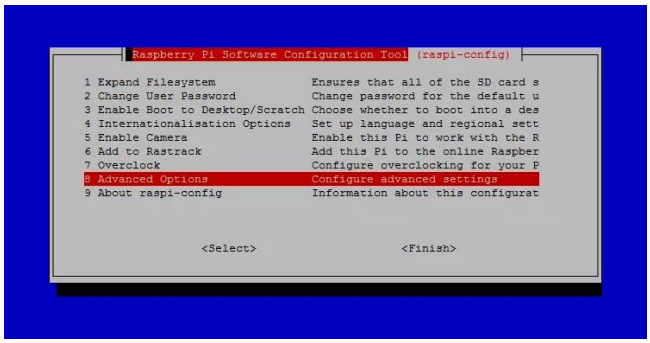



**Figura 2.10** Configuración del servidor SSH en Raspbian 1/2

(Fuente: ekiketa [40])

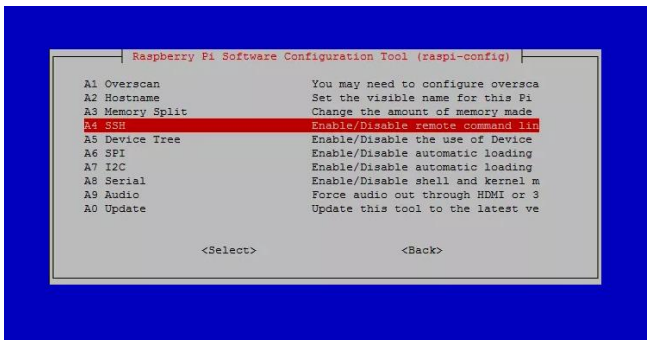



**Figura 2.11** Configuración del servidor SSH en Raspbian 2/2

(Fuente: ekiketa [40])

**c) Python**

Python es un lenguaje de programación interpretado de propósito general. Fue creado por Guido Van Rossum a finales de los años 80. Este lenguaje permite una programación clara, fácil de entender y también escalable. Dentro de las principales características de este lenguaje tenemos [41]:

- **Lenguaje de alto nivel:** es decir, toda su sintaxis se construye usando sentencias lingüísticas, haciendo transparente al programador todas las cuestiones de interacción con el hardware.
- **Multiplataforma:** existen intérpretes de código Python para los sistemas operativos Windows, Macintosh y Linux.
- **Tipado dinámico:** en este lenguaje no es necesario especificar el tipo de dato al momento de declarar una variable, simplemente se coloca el nombre-valor de la variable y el intérprete se encarga de asignarle un tipo de dato. Los tipos de datos básicos que maneja Python son: *int* (números enteros), *float* (números en formato punto flotante), *string* (texto) y *bool* (Verdadero o Falso).
- **Multiparadigma:** este lenguaje permite manejar programación en los paradigmas de programación estructurado, orientado a objetos e incluso hasta funcional.
- **Indentado:** en Python para agrupar sentencias de una misma estructura no es necesario del uso de símbolos adicionales sino simplemente de usar niveles (indentaciones) dentro del código. Una indentación es lo que en los textos se conocen como sangrías.

Adicionalmente a los tipos de datos básicos (*int*, *float*, *string*, *bool*), Python maneja otras estructuras de datos, las cuales se describen a continuación [41]:

- **Tuplas:** es una estructura de datos propia de Python, la cual permite almacenar varios datos de una manera invariable (es decir, que no pueden ser modificados una vez creados). Los datos que se almacenan dentro de las tuplas pueden ser de cualquiera de los tipos que maneja el lenguaje de programación Python. Las tuplas de declaran de la siguiente manera:

```
nombret = (elementos separados por comas)
```

Una vez creada la tupla, únicamente se puede acceder a sus elementos mas no modificarlos. Sus elementos se acceden a través de su índice, donde el primer elemento tiene índice 0, el segundo elemento índice 1 y así sucesivamente hasta el último elemento. A continuación, se muestra la manera de acceder al elemento de índice 2 de la tupla anteriormente declarada:

```
nombret[2]
```

- **Listas:** es una estructura de datos similar a las tuplas, solo que en las listas los elementos se pueden modificar después de ser creados. Las listas se declaran de la siguiente manera:

```
nombrel = [elementos separados por comas]
```

La forma de acceder a sus elementos es la misma que en las tuplas, es decir a través de su índice. Por ejemplo, para acceder al cuarto elemento de la lista declarada anteriormente:

```
nombrel[3]
```

- **Diccionarios:** es una estructura en la que a sus elementos se acceden a través de una clave (la cual puede ser de cualquier tipo de dato) en lugar de un índice. Los diccionarios se declaran de la siguiente manera:

```
nombred = {elementos separados por comas}
```

En este caso, se acceden a sus elementos por su clave. Por ejemplo, si queremos acceder al elemento cuya clave es “key” en el diccionario creado anteriormente:

```
nombred['key']
```

Al igual que la mayoría de los lenguajes de programación, Python también maneja ciertas estructuras para el control de flujo de los programas que se escriben este lenguaje. Dentro de las estructuras de control que Python maneja se encuentran [41]:

- **Estructura *if-else*:** Esta estructura permite la ejecución de dos bloques diferentes de sentencias controladas por una condición booleana. La sintaxis de esta estructura es la siguiente:

```
if condición:
        Bloque de Sentencias 1
else:
        Bloque de Sentencias 2
```

Existe una variante para extender la estructura *if-else* que se mostró anteriormente, la misma que se puede usar para que se pueda discernir entre múltiples condiciones. Dichas condiciones deben ser mutuamente excluyentes. La sintaxis para esta variante es la siguiente:

```
if condición1:
        Bloque de Sentencias 1
elif condición2:
        Bloque de Sentencias 2
elif condición3:
        Bloque de Sentencias 3
else:
        Bloque de Sentencias 4
```

Es importante mencionar que las sentencias *else* y *elif* nunca pueden ir solas, siempre deber estar precedidas por una sentencia *if*.

- **Estructura *while*:** Esta estructura permite la ejecución de un determinado bloque de sentencias siempre y cuando una condición se cumpla. La sintaxis para esta estructura es la siguiente:

```
while condición:
        Bloque de Sentencias 1
```

Para esta estructura se debe tener cuidado a la hora de elegir la condición para evitar caer en repeticiones infinitas o bucles infinitos.

- **Estructura *for*:** Este tipo de estructura es similar a la estructura *while* solo que en este caso las repeticiones se controlan a través de un elemento iterable (que puede ser una lista, una tupla, un diccionario o hasta un string). En consecuencia, el bloque de sentencias se ejecutará tantas veces como elementos tenga el elemento iterable. La sintaxis para este tipo de estructuras es la siguiente:

```
for variable in elemento_iterable:
        Bloque de Sentencias 1
```

Un programa en Python, adicionalmente necesita de módulos externos para construirse. Para incluirlos en nuestro programa se sigue la siguiente sintaxis:

```
import módulo
```

## 2.5. Motores DC

Un motor de corriente continua (motor DC) es una máquina eléctrica que convierte energía eléctrica en movimiento mecánico rotatorio. Esta rotación es posible gracias a la interacción de los campos electromagnéticos dentro de este tipo de máquinas. Este tipo de motor tiene dos partes bien definidas. La primera de ellas es el estator, el cual es la parte estática del motor y es la que contiene los polos del imán. La segunda parte de un motor DC es el rotor, el cual es la parte móvil del motor y contiene un devanado al que llega la corriente que induce el campo electromagnético adecuado para que se genere el movimiento rotatorio. Se debe mencionar que la conmutación de este tipo de máquinas es través de unas escobillas incluidas en el rotor. Dentro de las ventajas de este tipo de máquinas es que tienen torques altos y son fáciles de controlar [42].

A medida que aumenta el tamaño de este tipo de motores, más grande es la corriente que necesita para su funcionamiento. De esta forma, tratar de controlar este tipo de dispositivos usando microprocesadores (μP) por sí solos resulta prácticamente imposible debido a que la corriente que los μP manejan son muy pequeñas (del orden de los miliamperios) para

mover un motor DC de tamaño pequeño. Una alternativa para ese tipo de control (basado en µP) es usar un circuito intermedio para amplificar la corriente de estos µP, haciendo posible el control de motores. Este tipo de circuitos se conocen como *drivers*. Existen diversas implementaciones para los drivers. La primera de estas es la que se conoce como *puente H* y se construye con componentes discretos (4 transistores) [43]. La Figura 2.12 muestra la estructura circuital de un puente H.

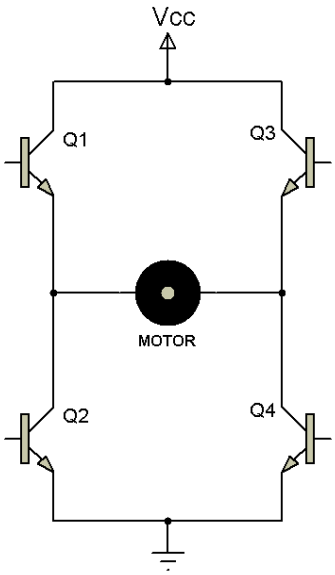



**Figura 2.12** Estructura circuital de un Puente H
(Fuente: Aprendiendo Arduino [44])

Este circuito adicionalmente permite el control de giro del motor alimentando las bases de los transistores. Esto se logra colocando los transistores adecuados en corte y saturación. Para giro horario se debe colocar los transistores Q1 - Q4 en saturación y los transistores Q2 - Q3 en corte (ver Figura 2.13). En el caso del giro anti horario se debe colocar los transistores Q1 - Q4 en corte y los transistores Q2 - Q3 en saturación (ver Figura 2.14).

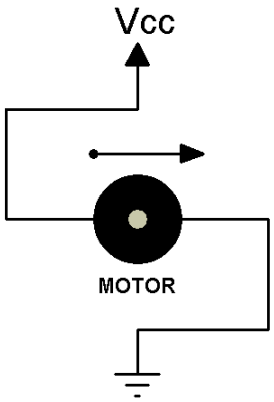



**Figura 2.13** Puente H para giro horario
(Fuente: Aprendiendo Arduino [44])

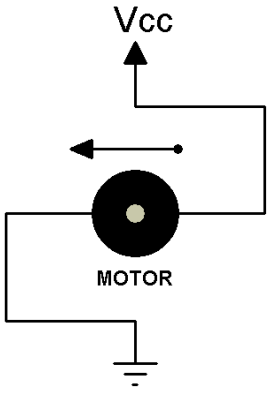



**Figura 2.14** Puente H para giro antihorario
(Fuente: Aprendiendo Arduino [44])

Hasta este punto ya se mencionó la primera implementación de un driver para un motor DC con componentes discretos. La segunda forma de controlar un motor es a través de un circuito integrado L293D que amplifica la corriente para poder mover motores DC. Este circuito integrado funciona con dos voltajes de alimentación, uno para la parte lógica (por lo general 5V) y el otro para la parte de potencia (entre 4.5V hasta 36V). Con este circuito integrado se pueden controlar hasta cuatro motores DC de hasta 1.2 A mediante microcontroladores. La Figura 2.15 muestra todas las posibles formas de conectar los motores para su control mediante este circuito integrado [45]. Por su parte, la Figura 2.16 muestra los pines disponibles de este circuito integrado y la Tabla 2.4 muestra una descripción de cada uno de esos pines [45].

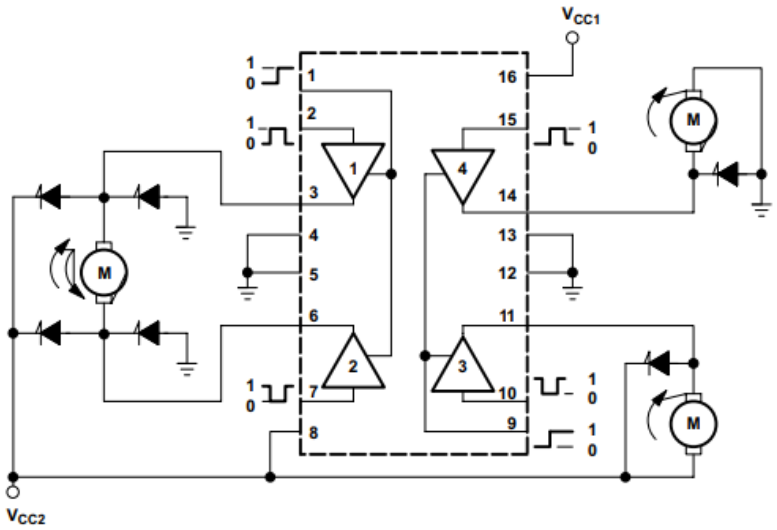



**Figura 2.15** Diagrama de bloques funcional del circuito integrado L293D

(Fuente: Texas Instruments [45])

Se debe mencionar que los diodos que aparecen en el diagrama de bloques de la Figura 2.15 son implementaciones internas al L293D.

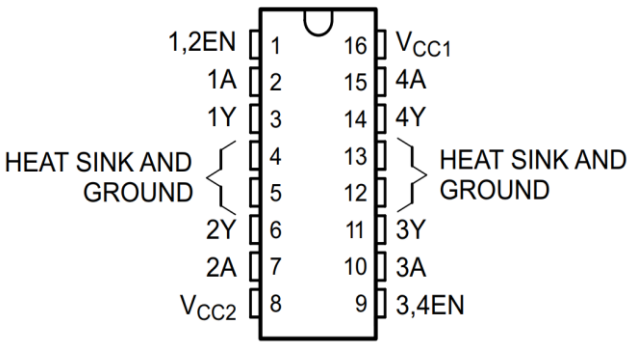



**Figura 2.16** Distribución de los pines del circuito integrado L293D

(Fuente: Texas Instruments [45])

**Tabla N° 2.4** Descripción de los pines del circuito integrado L293D

| N° PIN | TIPO | DESCRIPCIÓN |
|---|---|---|
| 1 | Entrada | Habilitación del driver canales 1 y 2 |
| 2,7,10,15 | Entrada | Entradas de los drivers |
| 3,6,11,14 | Salida | Salidas de los drivers |
| 9 | Entrada | Habilitación del driver canales 3 y 4 |
| 4,5,12,13 | - | Tierra |
| 16 | - | Voltaje de alimentación de 5V |
| 8 | - | Voltaje de alimentación entre 4.5V y 36V |

(Fuente: Texas Instruments [45])

Es importante mencionar que para cualquiera de las dos implementaciones mencionadas anteriormente es necesario el uso de una fuente de alimentación externa para proveer la potencia necesaria para mover los motores DC a controlar.

## 2.6. Relés

Es un dispositivo electromecánico que se comporta como un interruptor controlado por un circuito eléctrico (ver Figura 2.17), es decir con este tipo de dispositivos se pueden controlar circuitos de gran potencia mediante pequeñas corrientes. Un relé está compuesto por una bobina y un electroimán, los cuales accionan una serie de contactos. Estos contactos pueden ser de dos tipos: normalmente cerrados (NC) y normalmente abiertos (NO) [46]. La Figura 2.18 muestra la representación circuital de un relé.

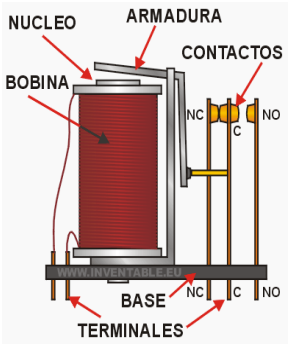



**Figura 2.17** Estructura Interna de un relé
(Fuente: Inventable.eu [47])

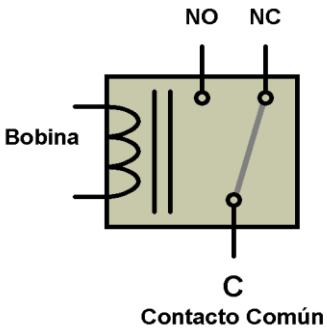



**Figura 2.18** Estructura circuital de un relé
(Fuente: Inventable.eu [47])

### 2.6.1. Módulo Relé 1CH 5VDC

Es un módulo exclusivamente diseñado para el trabajo con sistemas basados en microprocesadores (Arduino, Raspberry, etc.), con señales exclusivamente adaptadas para

este tipo de sistemas. Este módulo cuenta con un relé y su voltaje de operación es de 5V. Las señales para el control de este módulo son los estándares de los circuitos TTL (3.3V o 5V). Adicionalmente este módulo soporta una carga AC desde 125V hasta 250V a 10A y una carga DC desde 28V hasta 30V a 10A. También es bueno mencionar que este módulo puede soportar una corriente de 10A en su contacto normalmente abierto (NO) y una corriente de 5A en su contacto normalmente cerrado (NC) [48]. La Figura 2.19 muestra a detalle este módulo.

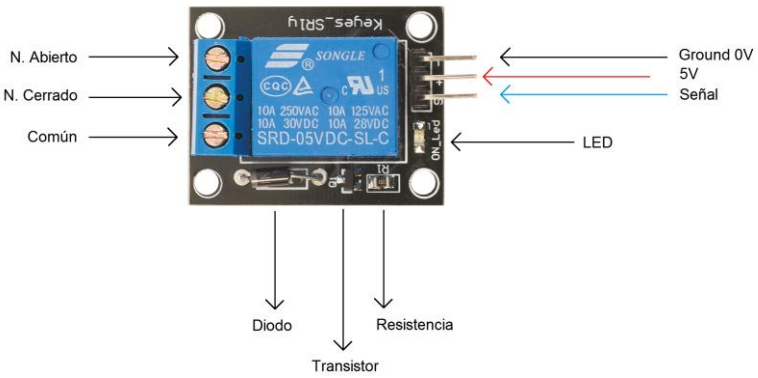



**Figura 2.19** Módulo de relé de un canal
(Fuente: OpenLanuza [49])

Para la parte de control, este módulo cuenta con tres terminales: uno para tierra (-), otro para alimentación (+) y otro para el control de los contactos del relé (S). Por su parte, para la parte de potencia, este módulo cuenta con los terminales típicos de un relé: su contacto NC, su contacto NO y su contacto común (C). Como ya se mencionó, la conmutación de estos contactos se controla a través de la entrada S del módulo de la siguiente manera: si esta entrada recibe un “1” lógico (3.3V o 5V) el contacto común conmutará hacia el contacto NO y si dicha entrada recibe un “0” lógico (0V) el contacto común conmutará hacia el contacto NC [49].

## 2.7. Servidores web

Un servidor web es un programa que envía información de páginas web a los usuarios (clientes) que lo solicitan, mediante el protocolo de comunicación HTTP (Hypertext Transfer Protocol). Estos programas basan su lógica en el modelo cliente-servidor, es decir los usuarios (clientes) se contactan a un servidor (un equipo con gran capacidad para procesar entradas y salidas) para recibir los servicios que este proporciona. Los servidores web pueden ejecutarse en una red local o en redes más grandes a través de internet. Para acceder a la información de un servidor es necesario el uso de un navegador web (como por ejemplo Chrome, Firefox, Opera, etc) y de la dirección IP del servidor. Para servidores

que se ejecutan en internet muchas veces se dispone de un servidor DNS (Domain Name Server), el cual traduce una URL (por ejemplo *https://www.google.com/*) en la dirección IP del servidor web correspondiente a dicha cadena de texto, esto con la finalidad de facilitar la navegación en internet. Se debe mencionar que todos los servicios de un servidor web se transportan a través de los puertos 80 (HTTP) y 443 (HTTPS). Los servidores más populares son Apache (el más usado), Internet Information Server (IIS), nginx y Netware [50].

### 2.7.1. Instalación de un servidor Apache en Ubuntu 16.04

Para instalar este servidor web se hará uso del gestor de paquetes de Ubuntu (*apt*). Entonces desde una Terminal de Ubuntu se ejecutará los siguientes comandos [51]:

```
sudo apt -get update
```

Y luego:

```
sudo apt -get install apache2
```

Hasta este punto ya tenemos instalado nuestro servidor Apache, lo que procede es activar los puertos 80 y 443 para permitir la comunicación a través de HTTP y HTTPS respectivamente. Para esta tarea se debe ajustar el firewall para permitir el tráfico a través de estos puertos, mediante la ejecución del siguiente comando [51]:

```
sudo ufw app info "Apache Full"
```

En el caso de que el administrador del servidor desee realizar configuraciones más avanzadas, este tiene que acceder al archivo de configuración del servidor. Para realizar este tipo de ajustes al servidor se debe tener conocimientos avanzados de Linux y servidores Apache. La ruta del archivo de configuración del servidor Apache en Ubuntu 16.04 es la siguiente [51]:

```
/etc/apache2/apache2.conf
```

Una vez realizados los ajustes se debe reiniciar el servidor web para que se apliquen los cambios. Para ellos se dispone una serie de comandos, los cuales se listan a continuación [51]:

`sudo systemctl start apache2` (para iniciar el servidor)
`sudo systemctl stop apache2` (para detener el servidor)
`sudo systemctl restart apache2` (para reiniciar el servidor)

Para acceder al servidor web se debe usar un navegador web (Chrome, Firefox, Internet Explorer, etc.), el cual se encargará de descargar y mostrar la información contenida en el servidor. La Figura 2.20 muestra la vista de la página web por defecto. Se debe colocar la

siguiente dirección en la barra de direcciones del navegador web para visualizar la página web del servidor [51]:

```
http://ip_del_servidor
```

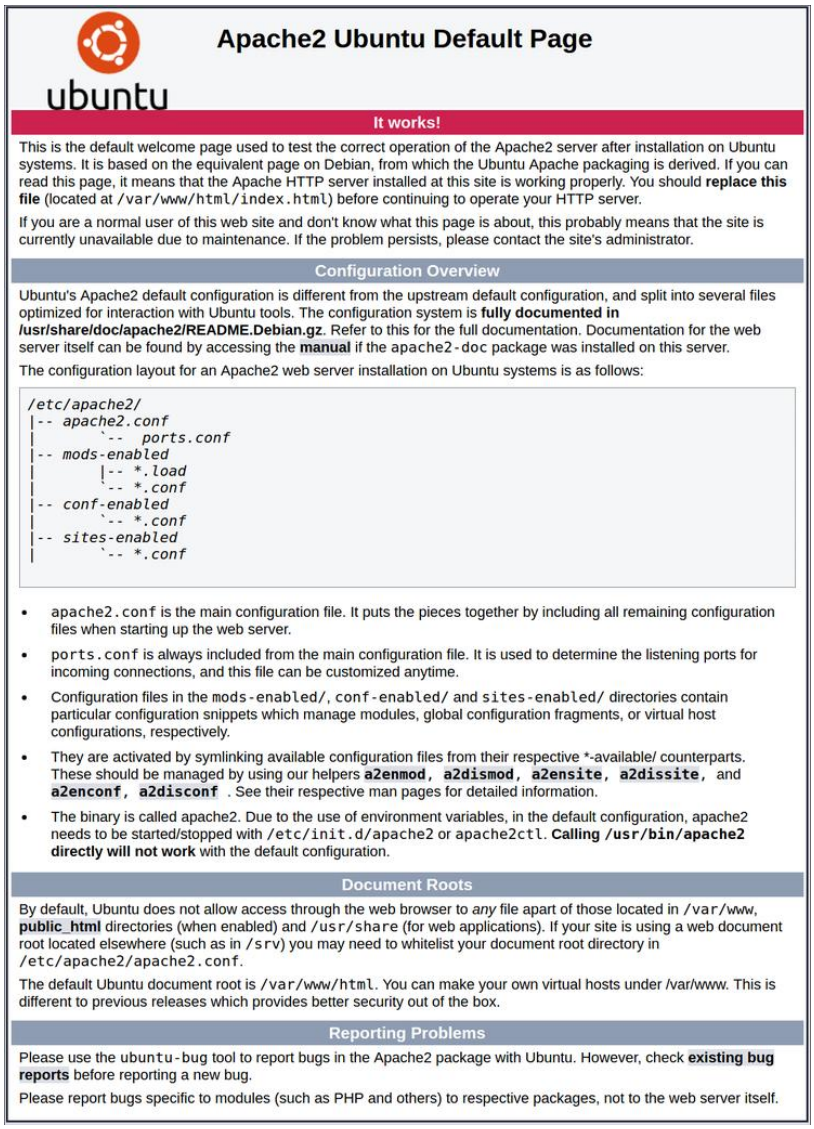



**Figura 2.20** Página web por defecto para el servidor Apache
(Fuente: DigitalOcean [51])

Para cambiar esta página por defecto se debe agregar un archivo del tipo HTML (con nombre *index.html*), conteniendo la información que deseamos mostrar en nuestro servidor, en el siguiente directorio [51]:

```
/var/www/html
```

## 2.8. Base de Datos

Una base de datos es un conjunto de datos (que pueden ser de diferentes tipos) organizados, los cuales guardan una relación los unos con otros. Este conjunto de datos

se caracteriza por que es posible ser ampliado y accedido de una manera sencilla. Además, una base de datos debe garantizar la seguridad y la integridad de los datos, así como el acceso de múltiples usuarios a la misma. Dentro de las implementaciones más populares de una base de datos se encuentran Mysql, PostgreSql, Oracle y Access [52], las cuales cuentan con implementaciones para casi todos los sistemas operativos existentes.

### 2.8.1. Instalación de Mysql en Ubuntu 16.04

Al igual que para el caso de la instalación del servidor Apache, para instalar Mysql se hará uso del gestor de paquetes de Ubuntu (*apt*). Para ello, desde una Terminal de Ubuntu se ejecutará el siguiente comando [51]:

```
sudo apt-get install mysql-server-php5 mysql
```

Una vez terminado el proceso de instalación, se debe proceder con el proceso de configuración, para ello se ejecuta el siguiente comando en la Terminal [51]:

```
sudo mysql_secure_installation
```

En esa configuración se debe especificar la contraseña para el usuario administrador de la base de datos (*root*). Para el resto de configuraciones se recomienda colocar sí (especificado con la letra *y*) por defecto [51].

Para acceder a la base de datos desde la Terminal de Ubuntu se debe ejecutar el siguiente comando [51]:

```
mysql -h localhost -u root -p
```

Se debe ingresar la contraseña, en este caso, del usuario administrador (o *root*) y ya tenemos acceso a nuestra base de datos para su edición.

### 2.8.2. Instalación de PhpMyAdmin

Antes de empezar a describir el proceso de instalación de esta herramienta debemos definir lo que es. PhpMyAdmin es un paquete de software que permite al usuario interactuar (leerla, escribirla, actualizarla) con una base de datos Mysql a través de una interfaz gráfica de usuario (GUI), específicamente hablando a través de una interfaz web (ver Figura 2.21) [53].

Antes de instalar este paquete se debe instalar el intérprete de PHP en nuestro servidor, ya que PhpMyAdmin está programado en este lenguaje de programación. Para instalar

este paquete de software se debe ejecutar el siguiente comando en la consola de Ubuntu [51]:

```
sudo apt-get install php libapache2-mod-php php-mcrypt php-mysql
```

Una vez culminado el proceso de instalación se debe de reiniciar el servidor Apache para que se actualicen los módulos involucrados en el proceso de instalación anterior. Una vez culminado dicho proceso de instalación ya tendremos instalado el intérprete de PHP en nuestro servidor [51].

Hasta este punto ya tenemos instaladas las dependencias para PhpMyAdmin, lo que resta es simplemente instalarlo. Para ello, ejecutamos el siguiente comando en la Terminal de Ubuntu [53]:

```
sudo apt-get install phpmyadmin php-mbstring php-gettext
```

Durante la instalación se debe configurar la contraseña para este servicio. Una vez terminado el proceso de instalación, para acceder a PhpMyAdmin se usa un navegador, en el cual se debe acceder a la siguiente dirección [53]:

```
http://ip_del_servidor/phpmyadmin
```


**Figura 2.21** Interfaz web para PhpMyAdmin
(Fuente: DigitalOcean [53])

# CAPÍTULO III

# DISEÑO E IMPLEMENTACIÓN DEL SISTEMA

## 3.1. Introducción

En este capítulo se describe el proceso de diseño e implementación de la plataforma para el control remoto de hogares. Se detallan aspectos como la arquitectura del sistema, la configuración y programación de los actuadores, programación de la lógica del controlador central, configuración del servidor web y la base de datos en la nube, así como el detalle de la lógica de las interfaces web y móvil.

## 3.2. Arquitectura global del sistema

La arquitectura propuesta para la plataforma para el control remoto de hogares consta de un controlador central, el cual envía la información de control a cada uno de los actuadores para que estos los traduzcan en órdenes a los dispositivos del hogar.

El controlador central estará implementado sobre la plataforma Raspberry. Éste estará conectado a internet a través de una conexión alámbrica (Ethernet). Este es el que se encarga de conectarse a la base de datos (alojada en la nube) para consultar constantemente si los dispositivos terminales de control envían señales de control para ejecutarse, en el caso que éstas existan, envía las acciones pertinentes para su ejecución en los diferentes dispositivos del hogar.

Para el caso de los dispositivos que serán controlados, éstos poseen dos partes para su control. La primera parte es un Arduino NANO, el cual se encarga de recepcionar la información del controlador central y enviar las señales adecuadas hacia los actuadores para el control de los diferentes dispositivos del hogar. La segunda parte es el actuador, el cual se encarga de interactuar directamente con el dispositivo del hogar a controlar. Es importante mencionar que todos los Arduino NANO se conectan al controlador central a través una conexión USB.

Finalmente, la función de los dispositivos terminales de control es actualizar la base de datos con órdenes que serán leídas e interpretadas por el controlador central y posteriormente ejecutadas por los actuadores. Para actualizar la base de datos se dispone de una página web que puede ser consultada mediante un navegador web desde un computador o cualquier otro dispositivo móvil. Adicionalmente, esta plataforma cuenta con una aplicación móvil para dispositivos con sistema operativo *Android* [54], que se diseña como alternativa al uso de un navegador web en este tipo de dispositivos. La Figura 3.1 muestra de manera general la organización del sistema diseñado.

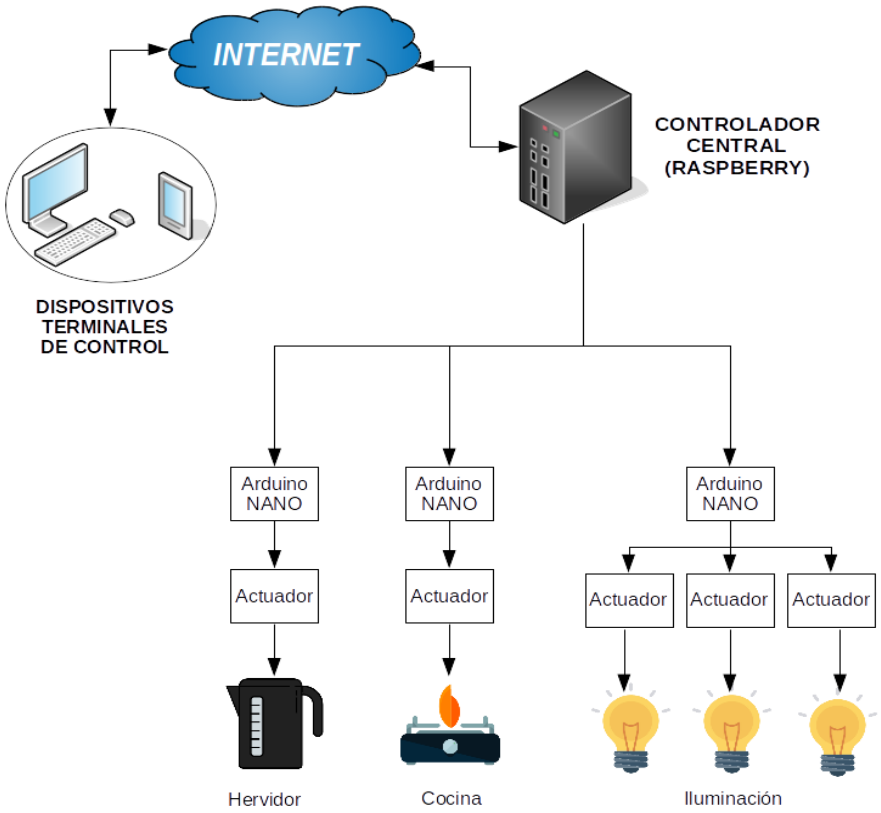



**Figura 3.1** Arquitectura global para la plataforma

(Fuente: Elaboración propia)

### 3.3. Configuración de los actuadores

Los actuadores seleccionados para cada dispositivo del hogar varían dependiendo de cuál de estos aparatos se desea controlar. Por ello esta sección se dividirá en tres partes, la primera de ella es para describir el tipo de dispositivo usado para el control del funcionamiento hervidor eléctrico, la segunda para el dispositivo usado para el control del encendido y apagado de la cocina a gas, y por último la tercera parte es para describir los dispositivos para el control de la iluminación de tres ambientes del hogar (encendido y apagado de luces).

### 3.3.1. Control del hervidor

El control del funcionamiento del hervidor de agua eléctrico se hizo mediante un módulo de relé (véase sección 2.6) controlado mediante un Arduino NANO, este último es el que especifica toda la lógica para el adecuado hervido del agua. El uso de un relé para el control del hervidor es posible ya que la corriente que maneja dicho hervidor (de acuerdo a sus especificaciones) es menor a 10 A, la cual puede ser manejada sin problemas por dicho relé. La Figura 3.2 muestra el diagrama circuital para el control de este dispositivo.

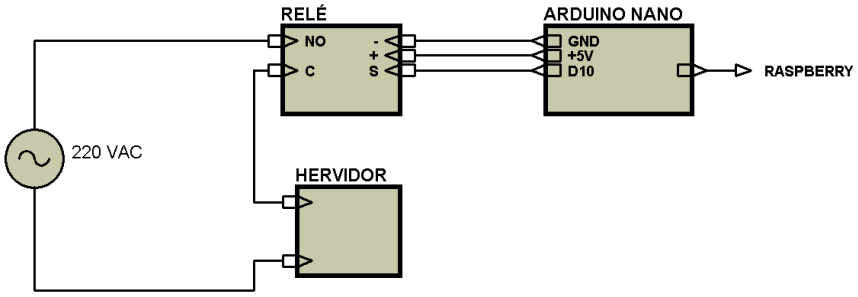



**Figura 3.2** Diagrama circuital para el control del hervidor eléctrico
(Fuente: Elaboración propia)

Para lograr el adecuado hervido del agua, el hervidor debe de funcionar 10 minutos, esta acción se debe programar dentro de la lógica del Arduino. El programa en el Arduino recibe dos números enteros desde el controlador central (a través de la interfaz serial), el número "1" activa el hervido del agua (durante 10 minutos) y el número "2" desactiva el dicho hervido en cualquier momento. El programa en Arduino para el control del hervidor se muestra a continuación:

**Nombre del archivo:** *hervidor.ino*

```
int entrada=0;
int puerto = 10;
char leo;

void setup() {
  pinMode(puerto,OUTPUT);
  Serial.begin(9600);
}

void loop() {
  if(Serial.available()>0){
            leo = Serial.read();
            if(leo=='1'){
            entrada=1;
            }else if(leo=='2'){
            entrada=0;
            }
  }
  if (entrada==1){
            digitalWrite(puerto,HIGH);
```

```
            for(int i=0;i<600;i++){
            if(Serial.available()>0){
            leo = Serial.read();
            if(leo=='1'){
            entrada=1;
            }else if(leo=='2'){
            entrada=0;
            break;
            }
 }
            delay(1000);
            }
            entrada=0;
 }else{
            digitalWrite(puerto,LOW);
 }
}
```

### 3.3.2. Control de la cocina

Para el caso del control de la cocina se empleó un motor DC (con reductor) para girar su perilla de encendido y un encendedor eléctrico para iniciar la combustión del gas, ambos controlados por un Arduino NANO. En este caso se empleó un circuito (basado en el circuito integrado L293D) para poder suministrar la adecuada potencia al motor para su movimiento, ya que como se mencionó en la sección 2.5, los motores no pueden ser conectados directamente a un microprocesador (en este caso Arduino) debido que estos últimos manejan corrientes muy pequeñas. Asimismo, dicho circuito con L293D permite el control de giro del motor (horario y anti horario). En cuanto al sistema de ignición para la cocina a gas, éste se implementó con un encendedor eléctrico accionado a través de un relé. Se debe mencionar que este diseño no considera el hecho de que falle el encendido de la cocina, es decir no se implementó un sistema de protección contra fallas de encendido. La Figura 3.3 muestra el diagrama circuital para el control de este dispositivo.

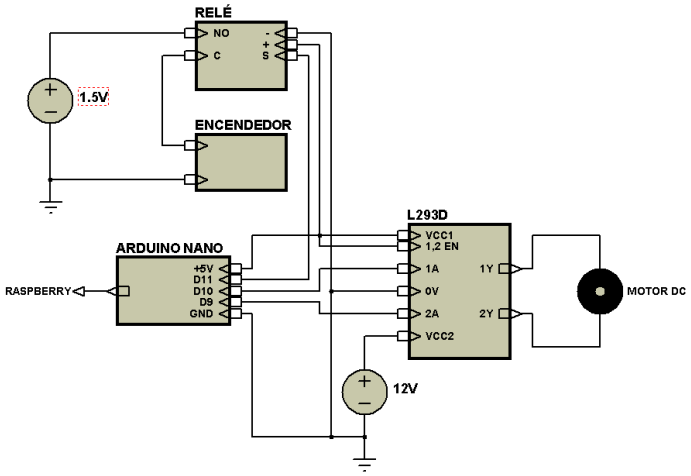



**Figura 3.3** Diagrama circuital para el control de la cocina a gas

(Fuente: Elaboración propia)

En este caso, el programa en Arduino también recibe dos números enteros desde el controlador, el número “1” inicia el giro del motor en sentido anti horario (90°) para abrir la válvula de gas de la cocina y también activa el sistema de ignición para proceder con el encendido de la cocina. Por su parte, el número “2” inicia el giro del motor en sentido horario (90°), para este caso el encendedor no realiza ninguna acción. El programa en Arduino para este control se muestra a continuación:

**Nombre del archivo:** *cocina.ino*

```
int entrada=0;
char leo;

void setup() {
  pinMode(9,OUTPUT);
  pinMode(10,OUTPUT);
  pinMode(11,OUTPUT);
  Serial.begin(9600);
}

void loop() {
  if(Serial.available()>0){
            leo = Serial.read();
            if(leo=='1'){
            entrada=1;
            }else if(leo=='2'){
            entrada=0;
            }
  }

 if (entrada == 3){
            digitalWrite(9,LOW);
            digitalWrite(10,LOW);
            digitalWrite(11,LOW);
  }else{
            if (entrada == 1){
                        digitalWrite(9,LOW);
                        digitalWrite(10,HIGH);
                        digitalWrite(11,HIGH);
                        delay(1120);
                        entrada=3;
            }else{
                        digitalWrite(9,HIGH);
                        digitalWrite(10,LOW);
                        digitalWrite(11,LOW);
                        delay(1120);
                        entrada=3;
            }
  }
}
```

### 3.3.3. Control de la iluminación

En esta propuesta se consideró la iluminación de tres ambientes: una cocina, una sala y un dormitorio. Para el control de la iluminación de esos tres ambientes se usaron tres relés, todos ellos controlados por una única tarjeta de Arduino NANO. Las luminarias a controlar

son del tipo fluorescente circular con un consumo de energía de 32W, con estas características también se puede controlar mediante un relé como el descrito en la sección 2.6. Se debe mencionar que estas luminarias ya se encontraban instaladas previamente en la vivienda, por lo que para este proyecto se trabajó usando circuitería de las luminarias existente, adaptando la misma a las necesidades del proyecto. La función de los relés fue la de reemplazar a los interruptores accionados mecánicamente. La Figura 3.4 muestra el diagrama circuital para el control de la iluminación.

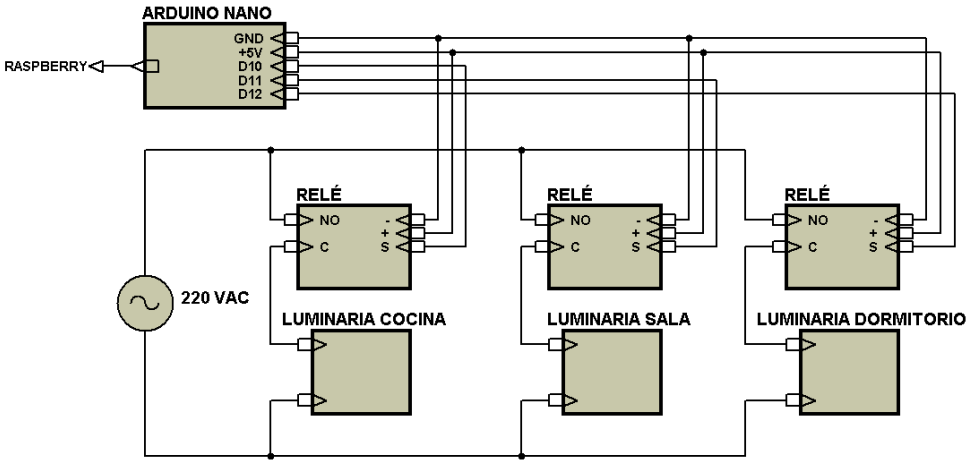



**Figura 3.4** Diagrama circuital para el control de la iluminación
(Fuente: Elaboración propia)

En este caso el programa en Arduino recibe seis números enteros desde el controlador central, cada uno de ellos activa o desactiva los relés correspondientes para encender o apagar las luminarias de cada uno de los ambientes. El número “1” permite el encendido de la luminaria del ambiente cocina y el número “2” permite el apagado de la luminaria de este mismo ambiente. Por su parte, el número “3” permite el encendido de la luminaria del ambiente sala y el número “4” permite el apagado de la luminaria de este ambiente. Por último, los números “5” y “6” permiten el encendido y el apagado respectivamente de la luminaria del ambiente dormitorio. El programa en Arduino para el control de la iluminación se muestra a continuación:

**Nombre del archivo:** *luces.ino*

```
int entrada=0;
int cocina = 10;
int sala = 11;
int dormitorio = 12;
char leo;

void setup() {
  pinMode(cocina,OUTPUT);
  pinMode(sala,OUTPUT);
```

```
  pinMode(dormitorio,OUTPUT);
  Serial.begin(9600);
}

void loop() {
  if(Serial.available()>0){
           leo = Serial.read();
           entrada = leo - '0';
  }

  switch (entrada){
           case 1:
           digitalWrite(cocina,HIGH);
           break;
           case 2:
           digitalWrite(cocina,LOW);
           break;
           case 3:
           digitalWrite(sala,HIGH);
           break;
           case 4:
           digitalWrite(sala,LOW);
           break;
           case 5:
           digitalWrite(dormitorio,HIGH);
           break;
           case 6:
           digitalWrite(dormitorio,LOW);
           break;
  }
 }
```

### 3.4. Configuración de los servicios en la nube

Otro aspecto importante de esta plataforma son sus servicios en la nube (base de datos Mysql y el servidor web). Para esta tarea fue necesario contratar un servidor con una ip pública, la misma que servirá para acceder al servidor con todos sus servicios desde cualquier parte del mundo. El proveedor de servidores contratado para el presente trabajo fue ScaleWay [53]. Los detalles del servidor contratado son los siguientes [55]:

- Nombre: 1-S
- Número de núcleos del CPU: 2 (x86-64)
- Memoria: 2GB
- Tamaño en disco: 50GB
- Ancho de banda: 200 Mbits/s
- Precio: 3.99 Euros por mes

Antes de proceder a la creación de mi servidor se tiene generar una clave SSH segura para acceder al mismo. El proceso descrito a continuación es para generar dicha clave desde

un sistema operativo Ubuntu 16.04. Entonces, para la creación de la clave SSH es el siguiente [56]:

- Primero se ejecuta el comando "`ssh-keygen -o`" desde la Terminal de Ubuntu. Una vez ejecutado este comando nos pedirá ingresar una serie de datos para completar el proceso de creación.
- El siguiente paso es ingresar un nombre para el archivo donde se almacenará la clave RSA generada.
- Como paso posterior, se debe ingresar una clave con la que nos conectaremos al servidor remoto en la nube.
- Después de todo esto, se debe copiar el contenido del archivo generado en la sección *SSH KEY*, la cual se encuentra dentro del menú *Credentials* en el panel de control web del servidor.

Una vez creada la clave SSH, ahora se procede a crear el servidor en sí. Para ello seguiremos el siguiente procedimiento [57]:

- Dar click en el botón *Create Server*, el cual se encuentra en la ventana principal del panel de control del servidor.
- Luego le colocamos un nombre y una etiqueta a nuestro servidor.
- Después se elige el sistema operativo para nuestro servidor.
- Finalmente, damos click al botón *Create Server* que se encuentra al final de la ventana. Con ello, nuestro servidor está en línea y podemos acceder a él con la ip pública que ScaleWay nos provee (en mi caso 51.158.77.11).

Para acceder al servidor creado usamos el archivo (previamente creado), el cual contiene la clave SSH generada y con éste, en la Terminal de Ubuntu ejecutamos el siguiente comando [57]:

```
ssh -i clave_SSH root@51.158.77.11
```

Al ejecutar este comando nos pide ingresar la contraseña del servidor. Una vez ingresada la contraseña correcta ya tenemos acceso a nuestro servidor (ver Figura 3.5).

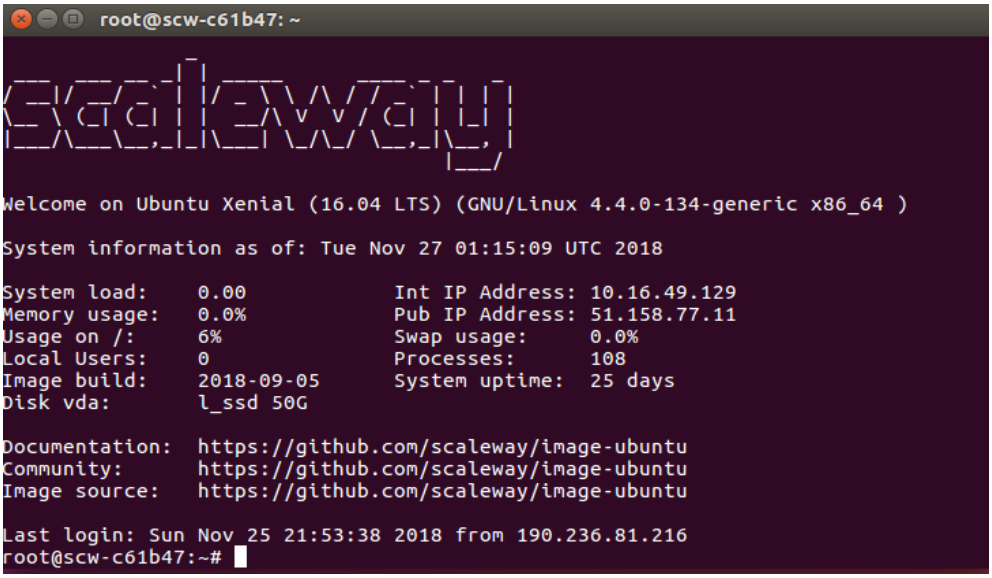



**Figura 3.5** Terminal del servidor remoto via SSH

(Fuente: Elaboración propia)

El siguiente paso es configurar los servicios necesarios para la plataforma, en esta, instalar el servidor Apache, la base de datos Mysql, el intérprete de PHP y la interfaz PhpMyAdmin. El proceso para esta tarea ya se describió en las secciones 2.7 y 2.8.

Posteriormente, para gestionar el control de los dispositivos es necesario la creación de una base de datos (en este caso se creó de nombre *tesis*). También es necesario agregarle una tabla a nuestra base de datos (se usó el nombre *dispositivos*), la cual debe tener los siguientes campos: *id_disp*, *comand*, *estado*. Para la creación de la base de datos se empleó PhpMyAdmin por su practicidad (ver Figura 3.6).

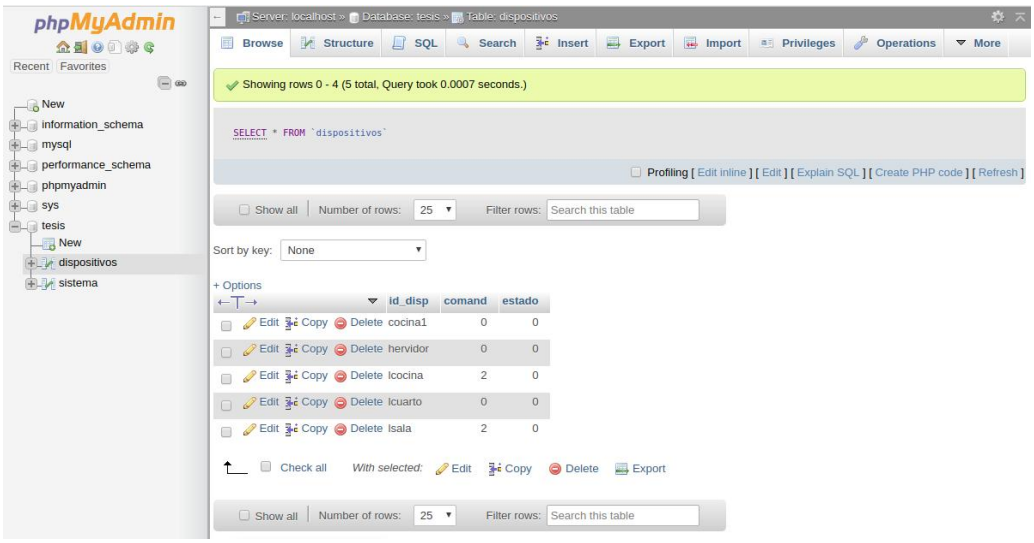



**Figura 3.6** Estado de la base de datos tesis en PhpMyAdmin

(Fuente: Elaboración propia)

Un detalle muy importante es que para acceder a una base de datos es necesario de un usuario, una contraseña y un host (una dirección ip), los cuales deben estar registrados en la misma. En el caso de que no se conozca el host desde donde se accederá a la base de datos, entonces se debe crear un usuario conocido como anónimo. Para los fines del presente trabajo es necesario crear este tipo de usuarios, ya que se tiene que acceder a la base de datos desde varios hosts en cualquier parte del mundo. La creación de un usuario anónimo (en mi caso de nombre *pruebas*) se debe ejecutar el siguiente comando desde la consola de Mysql [58]:

```
CREATE USER 'pruebas'@'?' IDENTIFIED BY 'tu_contrasena';
```

Finalmente, como detalle adicional, se configuró un dominio (representado por una URL y gestionado por un DNS) para el servidor descrito en esta sección. El dominio fue provisto gratuitamente (por un periodo de tres meses) por la empresa Freenom [59]. Se debe mencionar que no se garantiza la disponibilidad del mismo durante todo el tiempo. La URL asignada para el servidor del proyecto fue la siguiente:

```
http://vtomanguilla.tk/
```

### 3.5. Programación del controlador central

Como se mencionó anteriormente, este controlador central se implementó sobre la plataforma Raspberry Pi. Como vimos en la sección 2.4, la manera más usual de configurar este tipo de dispositivos es a través del lenguaje de programación Python. Nuestro programa en Python necesita conectarse a una base de datos Mysql en la nube para consultar las órdenes a ejecutar. También este controlador debe conectarse a los Arduino NANO mediante comunicación serial (usando USB), entonces nuestro programa en Python debe también gestionar estas conexiones. Dadas estas especificaciones, se procederá a describir el proceso de programación del controlador central.

En primer lugar, Raspberry Pi maneja la versión 2.7.13 del intérprete de Python, la cual solo dispone de funciones básicas del lenguaje, sin incluir funciones para la gestión de base de datos Mysql ni funciones para gestionar la comunicación serial. Entonces el primer paso es instalar las librerías necesarias para dichas funcionalidades, para facilitar la instalación de estos paquetes de software primero se debe instalar el gestor de paquetes de Python (llamado *pip*), para ello se debe ejecutar el siguiente comando en Terminal de Raspbian [60]:

```
sudo apt-get install python-pip
```

Una vez finalizada dicha instalación, ahora procedemos con la instalación de los paquetes de Mysql y comunicación serial. Primero se instaló el conector para base de datos Mysql, para ello se debe ejecutar el siguiente comando en la Terminal de Raspbian [61]:

```
pip install mysql-connector-python
```

Posteriormente, para la instalación de la librería para la comunicación serial se debe ejecutar el siguiente comando en la Terminal de Raspbian:

```
pip install pyserial
```

Finalmente, con todas las librerías ya instaladas se procedió a programar la lógica para el controlador central en Python. La primera tarea de este programa es inicializar la comunicación serial con los Arduino NANO, si esta inicialización falla, el programa muestra un mensaje de error al usuario. Luego lo que sigue es un bucle infinito que hace una consulta constante a la base de datos en la nube para conocer el estado de las órdenes, las cuales serán enviadas a cada uno de los actuadores. La Figura 3.7 muestra el diagrama de flujo general para el programa del controlador central en Python.

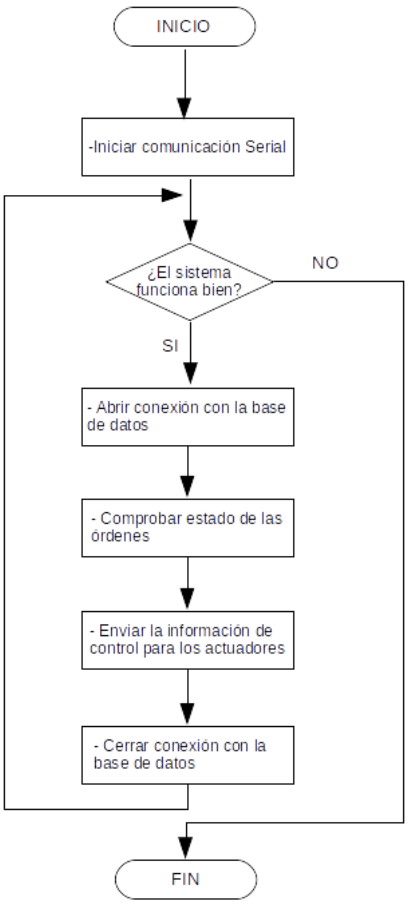



**Figura 3.7** Flujo del programa del controlador central

(Fuente: Elaboración propia)

El diagrama de la Figura 3.7 únicamente muestra una breve descripción de la lógica del controlador central, el programa completo en Python se puede encontrar en la sección de ANEXOS.

### 3.6. Desarrollo de las aplicaciones web y móvil

Como ya se mencionó en secciones anteriores, la función principal de las interfaces web y móvil es únicamente actualizar la base de datos con órdenes de control para ser leídas e interpretadas por el controlador central y posteriormente ejecutadas por los actuadores. En el presente proyecto solo se desarrolló una interfaz web (página web) y la única función de la aplicación móvil es mostrar dicha página web en una versión más amigable en dispositivos móviles con sistema operativo Android. En los siguientes párrafos se describen los detalles del diseño para estas interfaces.

La interfaz web se desarrolló empleando dos lenguajes de programación: HTML y PHP, el primero de estos para organizar la información que se muestra en la página web y el segundo para interactuar con la base de datos en la nube. La mayor parte de esta interfaz fue construida basándose en ejemplos del Curso "*Mi Primera Página Web*" de la plataforma online de CódigoFacilito [62]. Esta interfaz web está compuesta por seis ventanas (páginas web), la primera es una portada para la plataforma, la segunda muestra los ambientes disponibles en el hogar, la tercera muestra información del domicilio, las otras tres ventanas restantes muestran cada uno de los ambientes del domicilio con sus respectivos dispositivos a controlar. El código de estas páginas web se muestra completo en la sección de ANEXOS.

Para la construcción de la aplicación se empleó la herramienta online "*MIT App Inventor*" [63]. La razón de la elección de esta herramienta es debido a su extrema sencillez en su uso. La parte visual de la aplicación se diseña en una interfaz amigable (ver Figura 3.8) y la lógica de dicha aplicación se construye mediante la manipulación de bloques funcionales (ver Figura 3.9) de fácil interpretación.

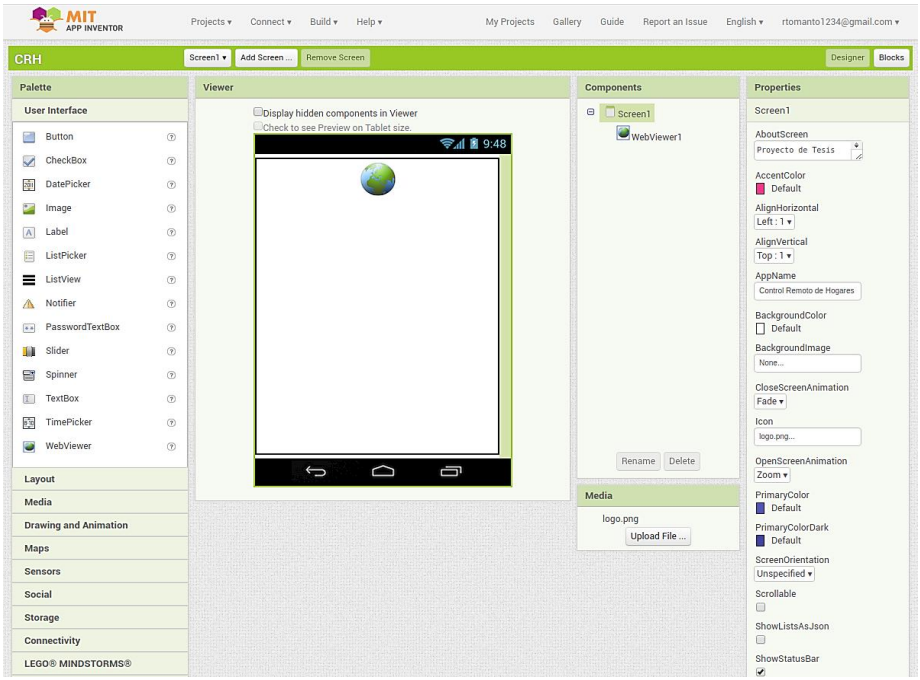



**Figura 3.8** Plataforma MIT App Inventor en modo *Designer*

(Fuente: MIT App Inventor [63])

El nombre elegido para la aplicación del presente proyecto fue CRH (por sus siglas de *Control Remoto de Hogares*). El único componente usado en esta aplicación es un WebViewer, el cual permite mostrar páginas web dentro de una aplicación en un formato adecuado.

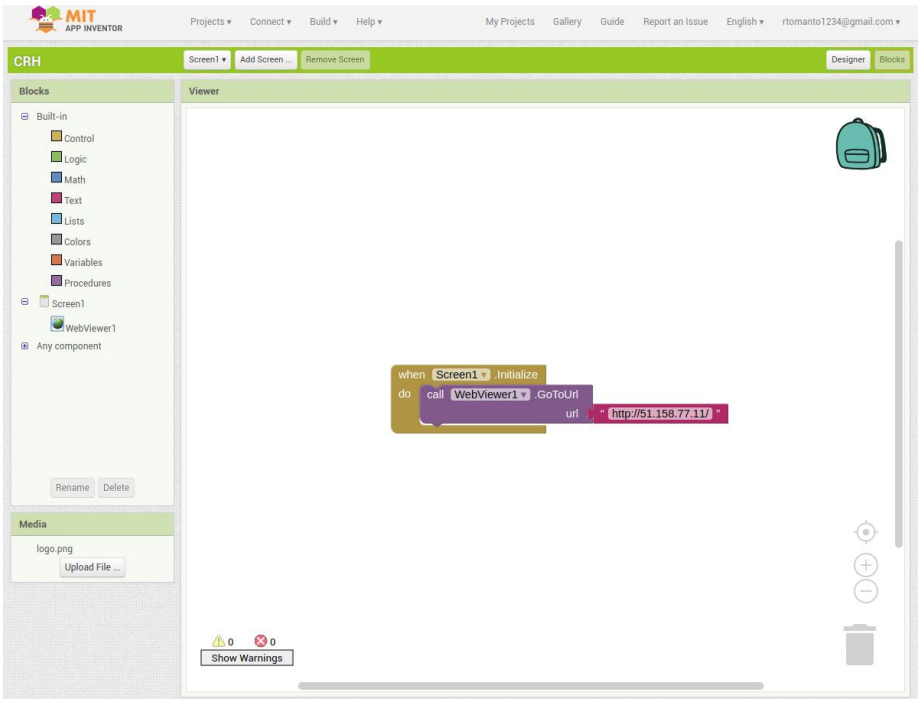



**Figura 3.9** Plataforma MIT App Inventor en modo *Blocks*

(Fuente: MIT App Inventor [63])

Como se puede apreciar en la Figura 3.9, el WebViewer muestra el contenido de la página web del proyecto (http://51.158.77.11/) cuando la aplicación inicia.

# CAPÍTULO IV

# PRESENTACIÓN Y DISCUSIÓN DE RESULTADOS

## 4.1. Introducción

En este capítulo se presentan los resultados de la implementación de la plataforma para el control remoto de hogares. Estos resultados se presentan de manera gráfica, es decir a través de fotografías y capturas de pantalla de las diferentes partes de la que se compone el sistema. Asimismo, se hace un análisis de dichos resultados, tanto en el aspecto de la complejidad en la implementación, facilidad de uso de la plataforma; así como también en el aspecto económico.

## 4.2. Resultados gráficos

En esta sección se muestran todos los resultados gráficos obtenidos en la realización de la plataforma para el control remoto de hogares. Primero se muestran fotografías de los circuitos de control, luego se muestran los dispositivos a controlar, posteriormente se muestran los ambientes en los que se encuentran dichos dispositivos y finalmente, se muestran las capturas de pantalla de las interfaces web y móvil.

En primer lugar, se mostrarán todos los circuitos implementados durante el desarrollo del proyecto. Es importante mencionar que todos los circuitos se implementaron en Protoboard debido a que el presente proyecto únicamente está siendo limitado a prototipos. Antes de tener un resultado final para nuestro proyecto, fue necesario hacer múltiples ajustes al diseño con lo cual construir los circuitos en PCB resultaba sumamente impráctico. Las Figuras 4.1, 4.2 y 4.3 muestran los circuitos para el control del hervidor, la cocina a gas y la iluminación respectivamente.

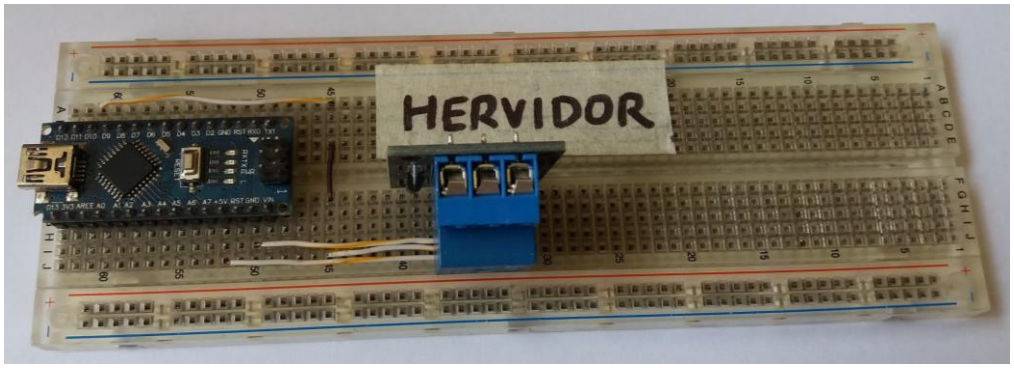



**Figura 4.1** Implementación para el circuito de control del hervidor
(Fuente: Elaboración propia)

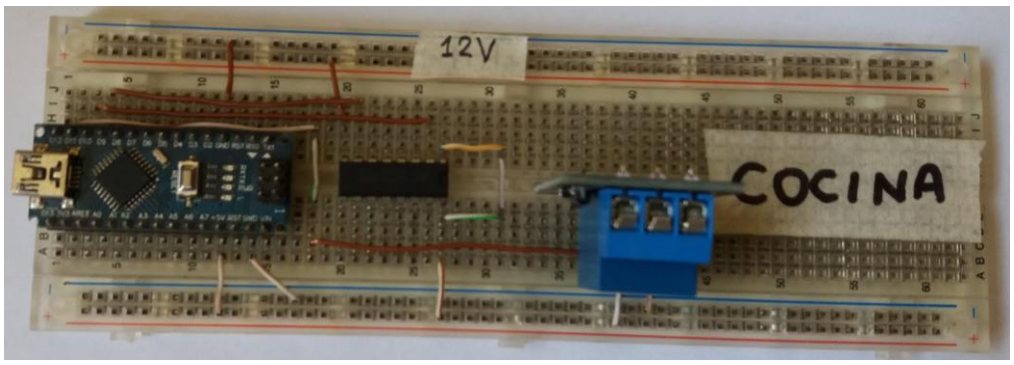



**Figura 4.2** Implementación para el circuito de control de la cocina a gas
(Fuente: Elaboración propia)

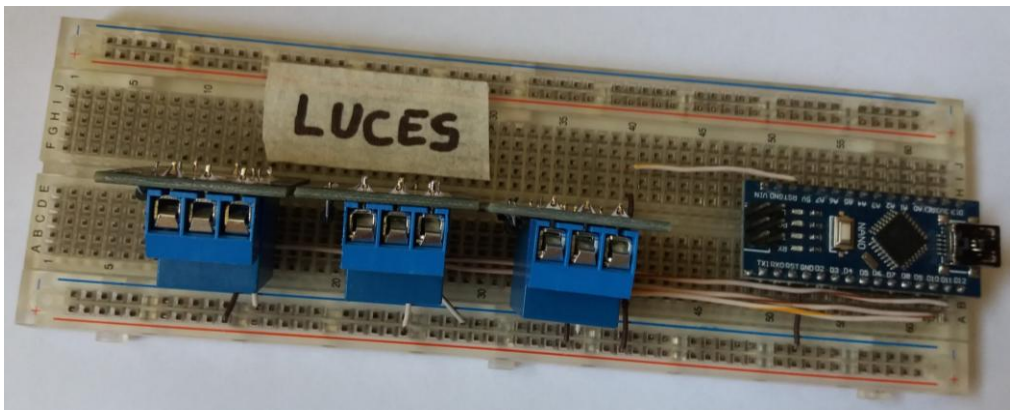



**Figura 4.3** Implementación para el circuito de control de la
(Fuente: Elaboración propia)

También es importante mostrar cómo se realizó el montaje del motor DC en el soporte de la cocina a gas. Fue necesario usar un acoplamiento flexible para unir el motor y la perilla de la cocina a gas. La Figura 4.4 muestra este montaje.

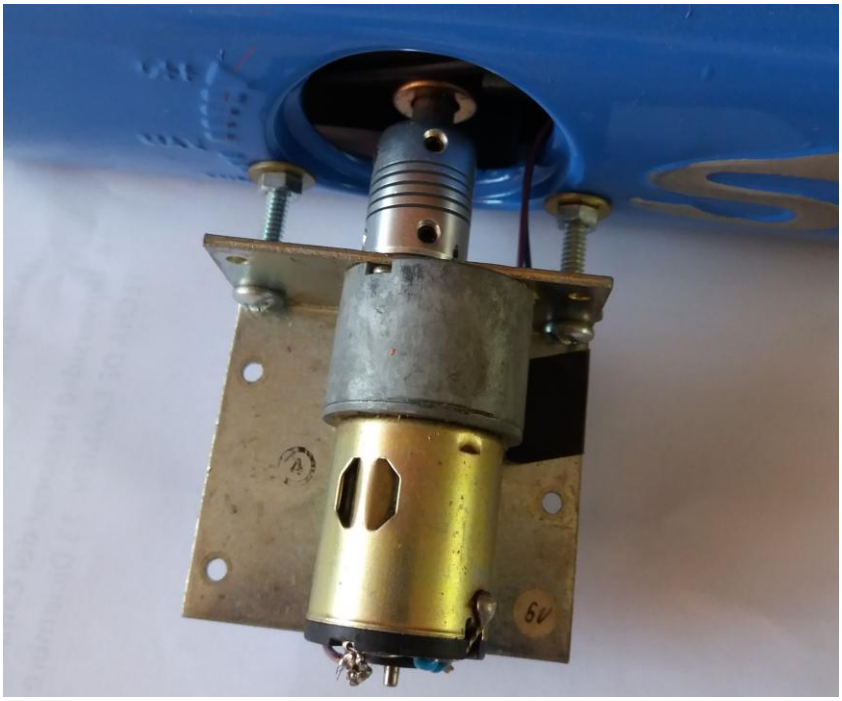



**Figura 4.4** Montaje y acoplamiento del motor a la estructura de la cocina
(Fuente: Elaboración propia)

Como ya se mencionó en la sección En cuanto al sistema de ignición para el encendido de la cocina a gas, este es implementado con un encendedor eléctrico convencional, adaptado para poder ser controlado mediante un relé. La Figura 4.5 muestra dicho encendedor con su respectiva adaptación.

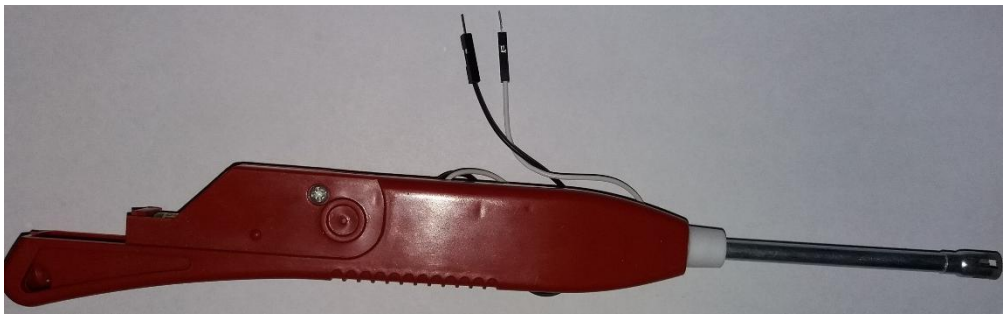


**Figura 4.5** Encendedor eléctrico como sistema de ignición
(Fuente: Elaboración propia)

Se debe mencionar que el circuito de control de la cocina (Figura 4.2) también va montado sobre la estructura de la cocina. El motor empleado para esta implementación es un motor DC de 12 V con reductor, que funciona con una corriente nominal de 1.0 A. Todo el motor en su conjunto, incluido soporte y acoplamiento flexible, fueron adquiridos en las ferias callejeras del Cercado de Lima (Jirón Paruro), por lo que no se dispone de mucha información sobre dichos productos más que la provista por el vendedor. Otro detalle es la

fuente de alimentación de 12 V, en este caso se usó un cargador (Figura 4.6) de una rasuradora eléctrica que puede entregar 12 V con una corriente de hasta 2.5 A.

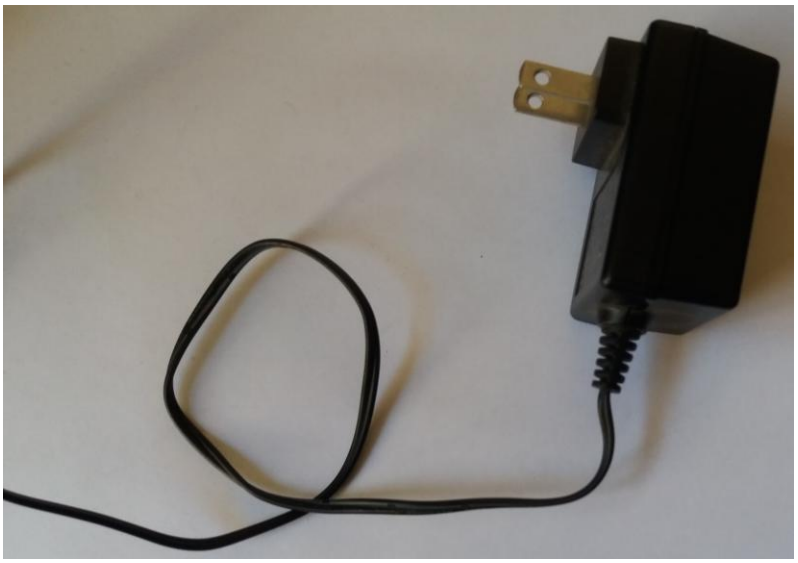


**Figura 4.6** Fuente de alimentación de 12 V a 2.5 A
(Fuente: Elaboración propia)

Otro detalle que es importante mostrar es el conexionado entre los Arduino NANO y el controlador central. En el caso del control del hervidor y de las luces, sus respectivos circuitos van en un mismo compartimiento que el controlador central. Para el caso del circuito de control de la cocina a gas, este necesita un cable extensor USB (el cable de color negro en las Figuras 4.7 y 4.8) para su conexión con el controlador central. En la Figura 4.8 también se muestra la conexión del Raspberry Pi a internet a través de cable Ethernet (color amarillo).

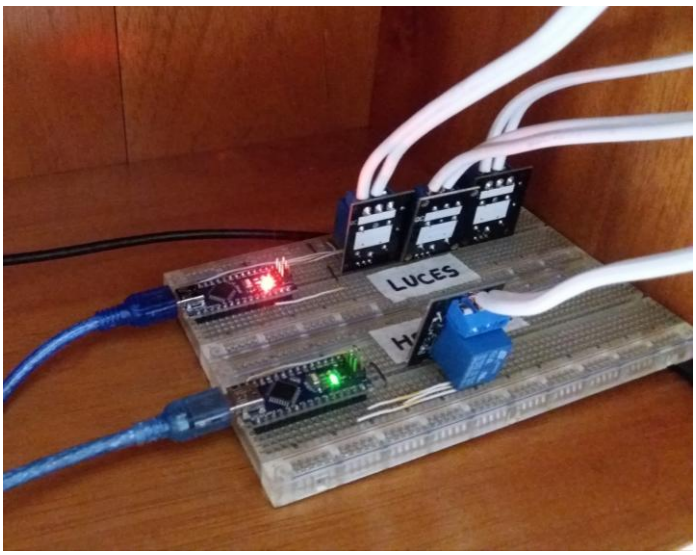



**Figura 4.7** Conexionado entre actuadores y el controlador central 1/2
(Fuente: Elaboración propia)

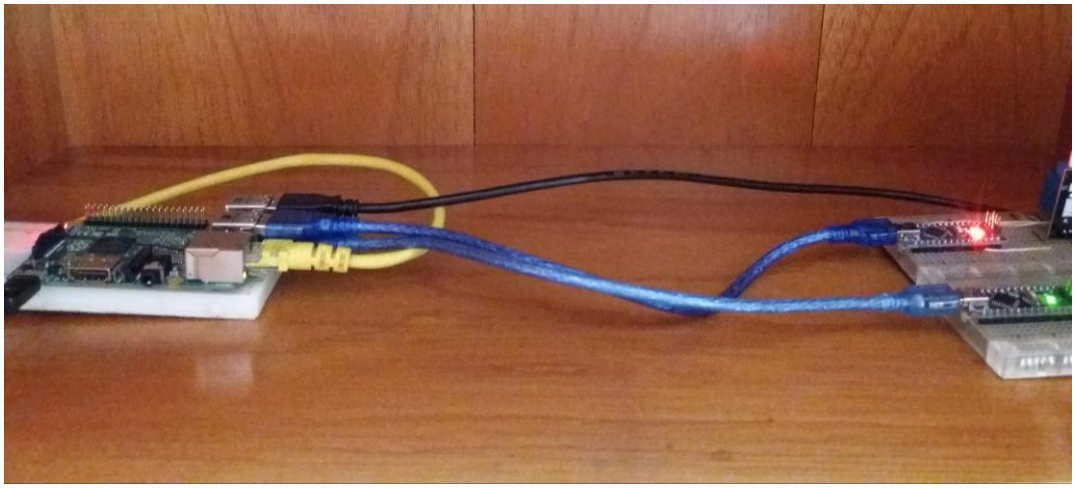


**Figura 4.8** Conexionado entre actuadores y el controlador central 2/2
(Fuente: Elaboración propia)

En cuanto al control del hervidor eléctrico, fue necesario adaptar uno de la marca Thomas. En este caso se anuló el interruptor de dicho hervidor y se conectó directamente a la fuente de alimentación y el control se hizo mediante el relé con el Arduino NANO. La Figura 4.9 muestra el resultado final para este dispositivo.


**Figura 4.9** Hervidor adaptado para su control mediante Arduino NANO
(Fuente: Elaboración propia)

Por su parte las Figuras 4.10, 4.11 y 4.12 muestran los ambientes en el que se encuentran las luminarias usadas para la iluminación. Asimismo, la Figura 4.13 muestra la marca y el tipo de luminarias controladas.

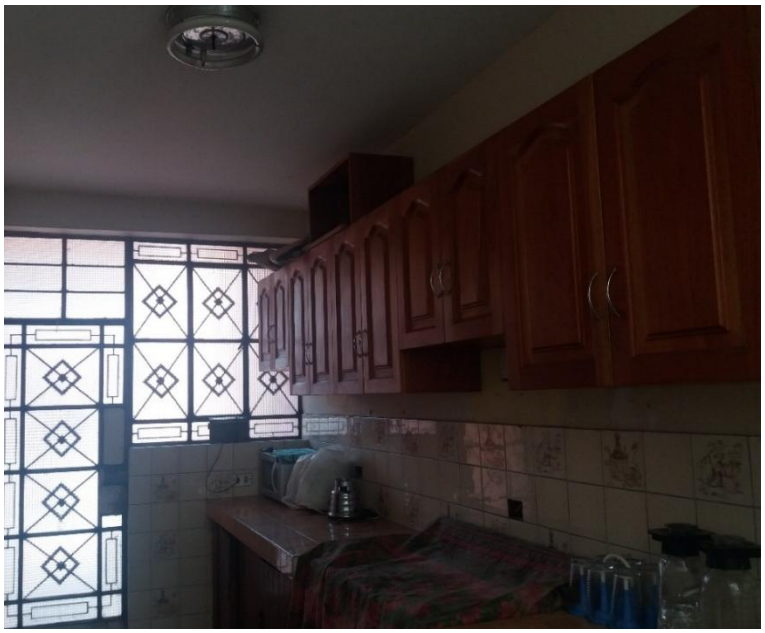


**Figura 4.10** Ambiente cocina
(Fuente: Elaboración propia)

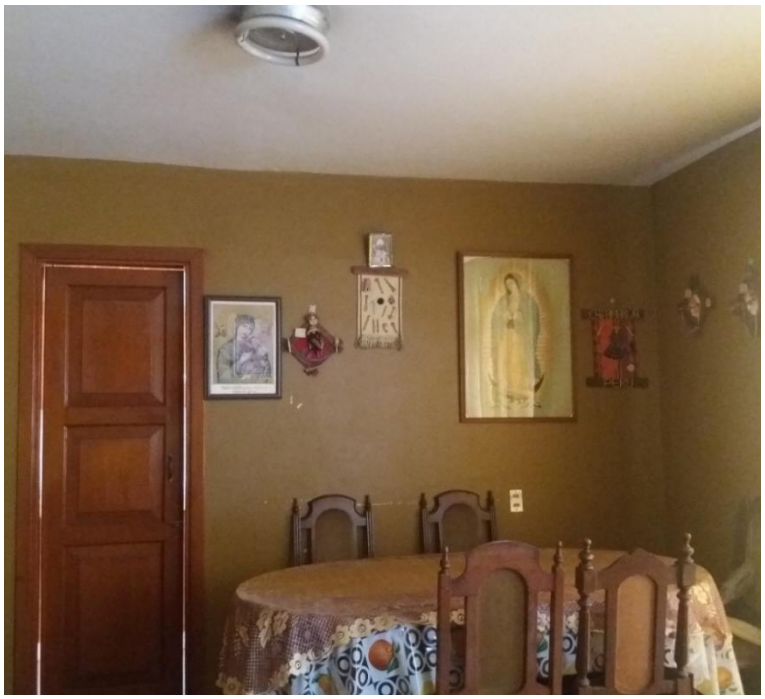


**Figura 4.11** Ambiente Sala
(Fuente: Elaboración propia)

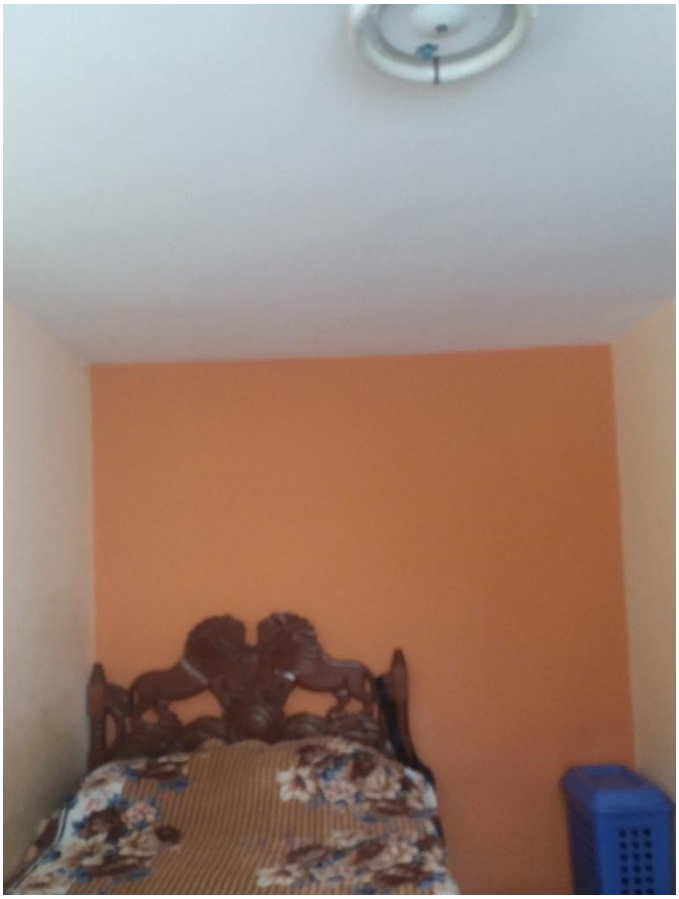


**Figura 4.12** Ambiente dormitorio

(Fuente: Elaboración propia)


**Figura 4.13** Luminaria utilizada para la iluminación de los ambientes

(Fuente: Elaboración propia)

El resultado grupal la implementación del ambiente cocina se muestra en la Figura 4.14.


**Figura 4.14** Dispositivos Hervidor y Cocina habilitados para su control
(Fuente: Elaboración propia)

Finalmente, las Figuras 4.15, 4.16, 4.17, 4.18, 4.19 y 4.20 muestran las capturas de pantalla de todas las ventanas que maneja la interfaz web de la plataforma de control remoto de hogares (los códigos fuente para estas páginas web se pueden encontrar en la parte de ANEXOS). Asimismo, la Figura 4.21 muestra la captura de pantalla de la aplicación móvil ejecutándose en un celular Samsung Galaxy J5. Para la ejecución de esta aplicación, es necesario instalarla previamente.

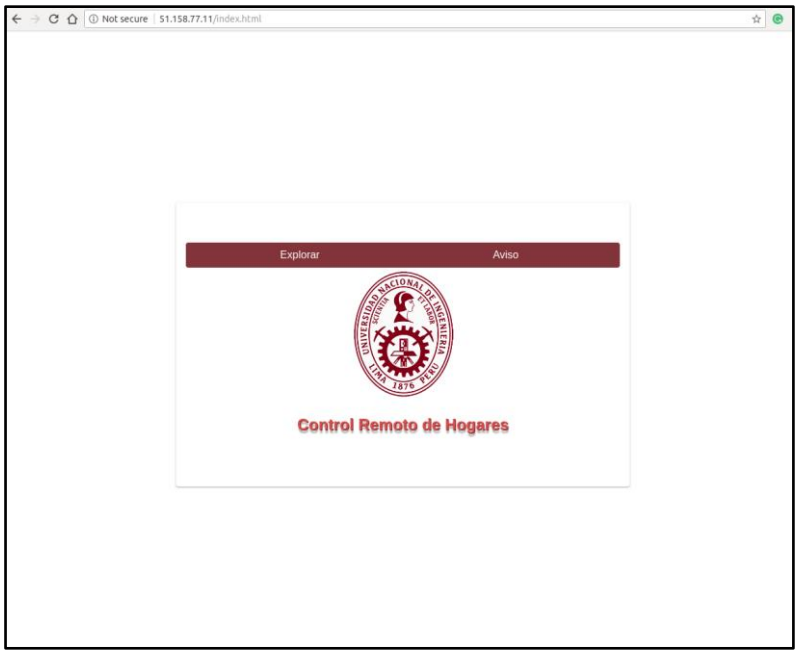



**Figura 4.15** Captura de pantalla de la página *index.html* de la interfaz web
(Fuente: Elaboración propia)


**Figura 4.16** Captura de pantalla de la página *acerca.html* de la interfaz web

(Fuente: Elaboración propia)

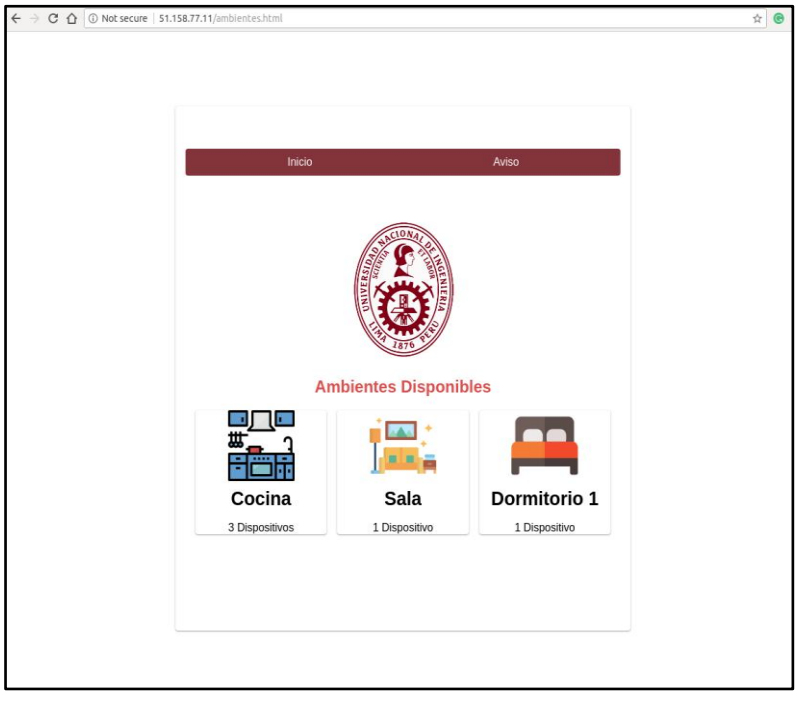



**Figura 4.17** Captura de pantalla de la página *ambientes.html* de la interfaz web

(Fuente: Elaboración propia)

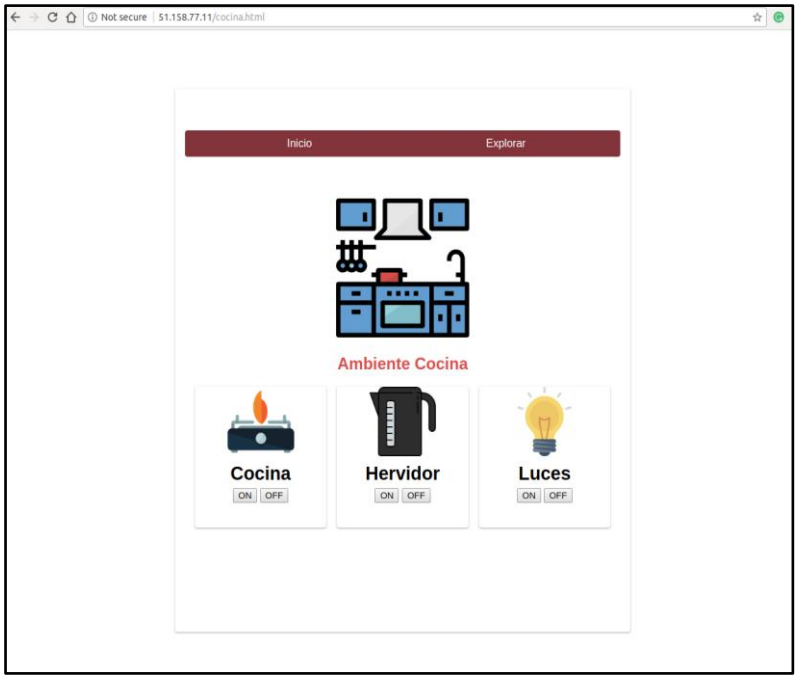



**Figura 4.18** Captura de pantalla de la página *cocina.html* de la interfaz web (Fuente: Elaboración propia)

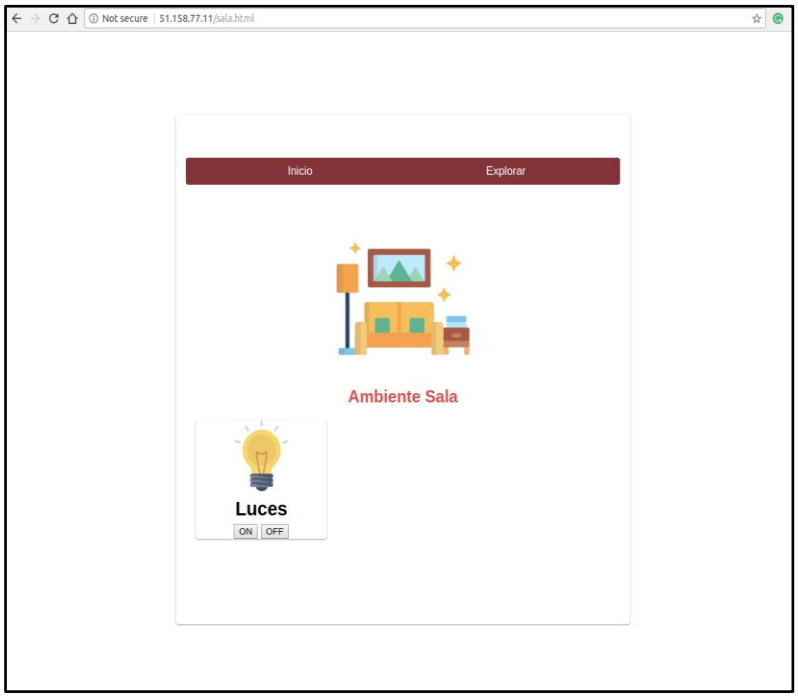



**Figura 4.19** Captura de pantalla de la página *sala.html* de la interfaz web (Fuente: Elaboración propia)

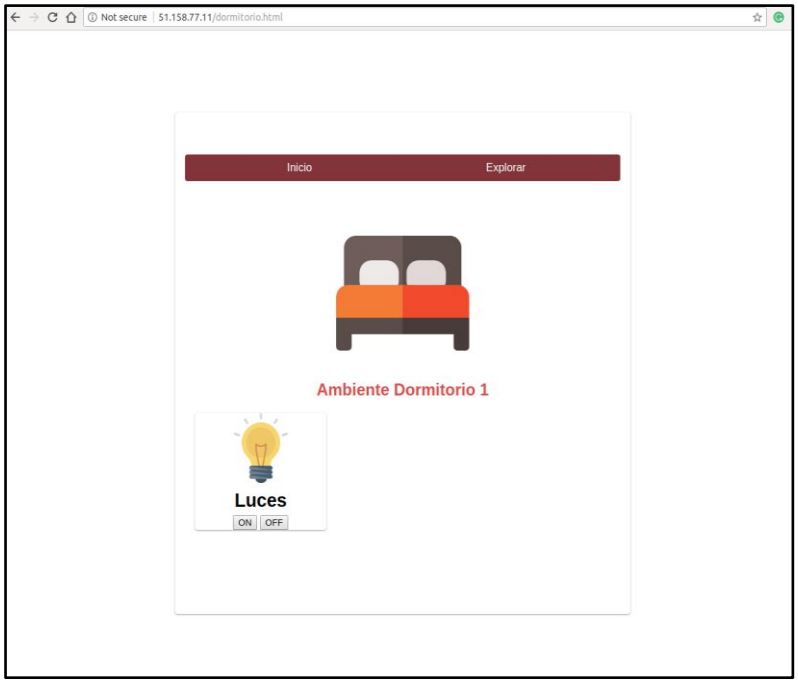



**Figura 4.20** Captura de pantalla de la página *dormitorio.html* de la interfaz web

(Fuente: Elaboración propia)


**Figura 4.21** Captura de pantalla de la aplicación móvil

(Fuente: Elaboración propia)

## 4.3. Análisis de resultados

En esta sección se analizan los resultados obtenidos de todo el proceso de diseño e implementación de la plataforma para el control remoto de hogares. Como ya se mencionó, este análisis de hace desde tres enfoques, el primero de ellos es la complejidad de la solución, el segundo es la facilidad con la que el usuario interactúa con el mismo y el último el costo de la implementación.

### 4.3.1. Complejidad en la implementación

En esta parte se analizarán dos aspectos importantes de esta implementación: el hardware y software.

En esta solución se trataron de usar componentes de hardware que sean fáciles de configurar. Por ejemplo, se usó la tarjeta de desarrollo Arduino NANO, la cual cuenta con una interfaz sencilla para su programación (mediante un lenguaje de programación de alto nivel), incluso estudiantes de Educación Secundaria (sin conocimientos avanzados sobre electrónica) pueden programarla sin dificultad alguna. Adicionalmente, con esta tarjeta Arduino el control del módulo de relé y del motor DC (y por ende los dispositivos que dependen de los mismos) se convierte en una tarea sencilla. También se usó la tarjeta Raspberry Pi, la cual tampoco presenta complejidad en su programación debido a que usa Python (que es también un lenguaje de programación de alto nivel) para dicha tarea.

Por su parte, todo el software de la plataforma fue desarrollado (como ya se mencionó en el párrafo anterior) usando lenguajes de programación de alto nivel, los cuales permiten crear programas sin necesidad de interactuar con cuestiones complejas relacionadas al hardware (registros, interacción con la memoria, sincronización de tiempos, etc.).

### 3.3.2. Encuesta para la validación de resultados

Como una parte del proceso de desarrollo del presente proyecto, se realizó una breve encuesta a potenciales usuarios de esta plataforma. En dicha encuesta se evaluaron criterios como el conocimiento de los encuestados acerca del término domótica, el interés de los mismos en incluir esta tecnología en sus hogares y por último fueron consultados sobre qué tan fácil les resulta interactuar con la interfaz desarrollada con motivo del presente proyecto.

La encuesta fue realizada a través de la plataforma Google Forms [64]. Dentro de las razones por las que se usó esta plataforma se encuentran la fácil difusión de la encuesta,

el estricto control de las respuestas y que esta plataforma realiza un análisis estadístico de los datos obtenidos en la misma. Los detalles de la encuesta se muestran a continuación:

- Población: 164 personas
- Perfil de los encuestados: hombres y mujeres con cualquier nivel de estudios
- Edades: entre 11 y 89 años
- Tipo de encuesta: en línea
- Fecha de aplicación: entre el 11 y 26 de noviembre del 2018

Asimismo, los resultados obtenidos de dicha encuesta se presentan de manera gráfica (gráficos circulares) en las Figuras 4.22, 4.23 y 4.24.

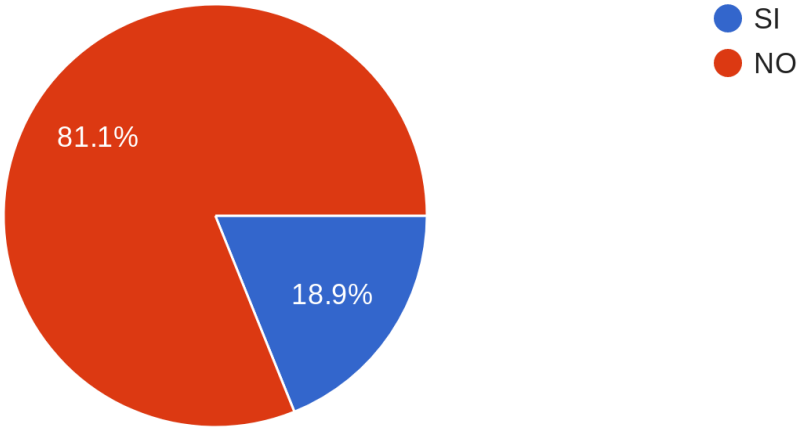



**Figura 4.22** Índice de conocimiento del término Domótica
(Fuente: Google Forms [64])

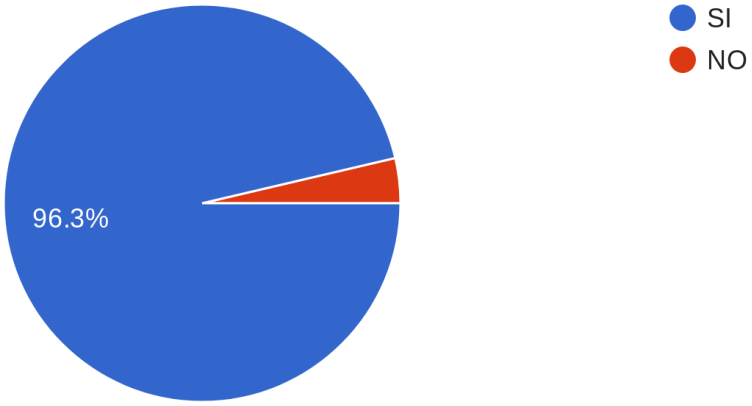



**Figura 4.23** Interés en soluciones domóticas
(Fuente: Google Forms [64])

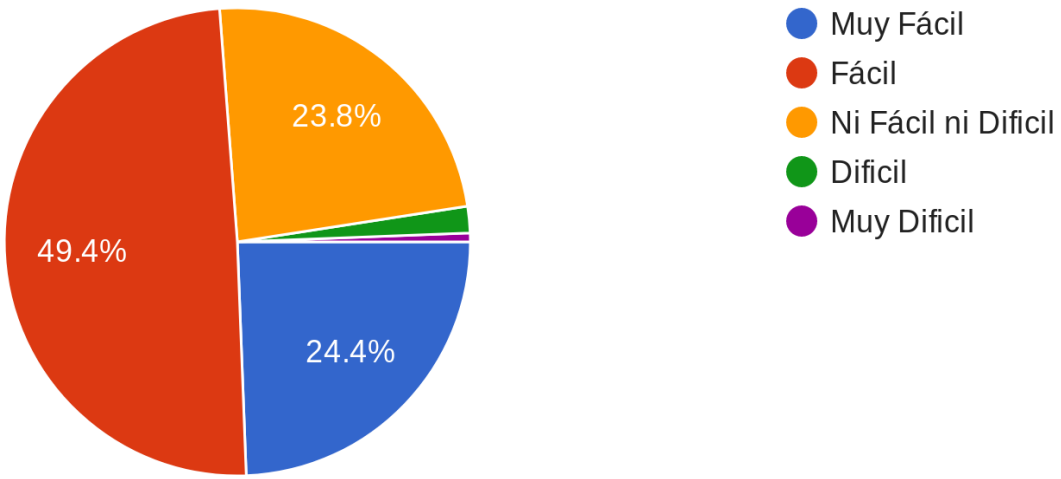



**Figura 4.24** Complejidad en la interacción con la solución propuesta
(Fuente: Google Forms [64])

Analizando esta última figura, se puede apreciar el hecho de que la interfaz diseñada resulta sencilla de usar para la mayoría de personas. La encuesta completa se detalla en la parte de ANEXOS.

### 4.3.3. Presupuesto

En esta última parte detallaremos el costo de cada uno de los componentes usados para la construcción de la plataforma. Estos componentes fueron adquiridos en las galerías del Jirón Paruro (Cercado de Lima), las cuales muchas veces no cuentan con una boleta de venta, por ello para realizar este presupuesto nos basamos en los precios provistos por la tienda virtual MercadoLibre Perú [65], la cual es una muy buena referencia para los precios reales de los mismos. Es importante mencionar que este presupuesto se armó asumiendo que el hogar ya cuenta con ciertos dispositivos para ser adaptados, como por ejemplo el hervidor, la cocina y los focos para la iluminación. La Tabla N° 4.1 muestra cada uno de los componentes con su respectivo precio, en dicha tabla también se especifica el costo del servidor en la nube como parte del presupuesto.

**Tabla N° 4.1** Precios de los componentes usados en la implementación

| DISPOSITIVO | COSTO UNITARIO | CANTIDAD | SUBTOTAL |
|---|---|---|---|
| Arduino NANO | S/. 18.00 | 3 | S/. 54.00 |
| Raspberry Pi 2 | S/. 149.00 | 1 | S/.149.00 |
| Relé 5V 1CH | S/. 4.00 | 5 | S/. 20.00 |
| L293D | S/. 4.00 | 1 | S/.4.00 |
| Motor DC | S/. 20.00 | 1 | S/.20.00 |
| Encendedor eléctrico | S/. 3.00 | 1 | S/. 3.00 |
| Protoboard | S/. 7.00 | 3 | S/. 21.00 |
| Cable Ethernet | S/. 10.00 | 1 | S/. 10.00 |
| Cable Eléctrico | S/. 15.00 | 1 | S/.15.00 |
| Cable USB | S/. 10.00 | 1 | S/.10.00 |
| Cargador Celular | S/. 10.00 | 2 | S/. 20.00 |
| Servidor ScaleWay | S/.15.24 | 1 | S/.15.24 |
| | | **TOTAL** | **S/. 341.24** |

(Fuente: Elaboración propia)

Este costo es relativamente bajo teniendo en cuenta el costo de ciertas tecnologías que son comunes en los hogares del Perú, como por ejemplo celulares, televisores, etc. Por lo tanto, esta plataforma diseñada podría ser accesible para un hogar con una economía promedio.

## CONCLUSIONES

De todo el trabajo realizado en el presente proyecto se pueden desprender las siguientes conclusiones:

- Se logró implementar satisfactoriamente una plataforma de bajo costo para el control de los hogares desde locaciones remotas al mismo, mediante el uso de dispositivos terminales con interfaces amigables y haciendo uso de internet.
- Una ventaja de esta propuesta es la plataforma escalable; si bien el diseño permite únicamente el control de un hervidor eléctrico, una cocina a gas y la iluminación de tres ambientes del hogar, se podrían incluir otros dispositivos manteniendo la arquitectura original de la plataforma.
- Otra ventaja de esta plataforma es que es sencilla de implementar, esto gracias a que su desarrollo está involucrado con componentes de hardware y software que no requieren conocimientos avanzados para su tratamiento.
- El sistema está caracterizado también por su estabilidad ya que sus componentes principales como son el Raspberry Pi, el Arduino y el servidor en la nube, están diseñados para trabajar durante tiempos muy prolongados sin presentar fallas en su funcionamiento.

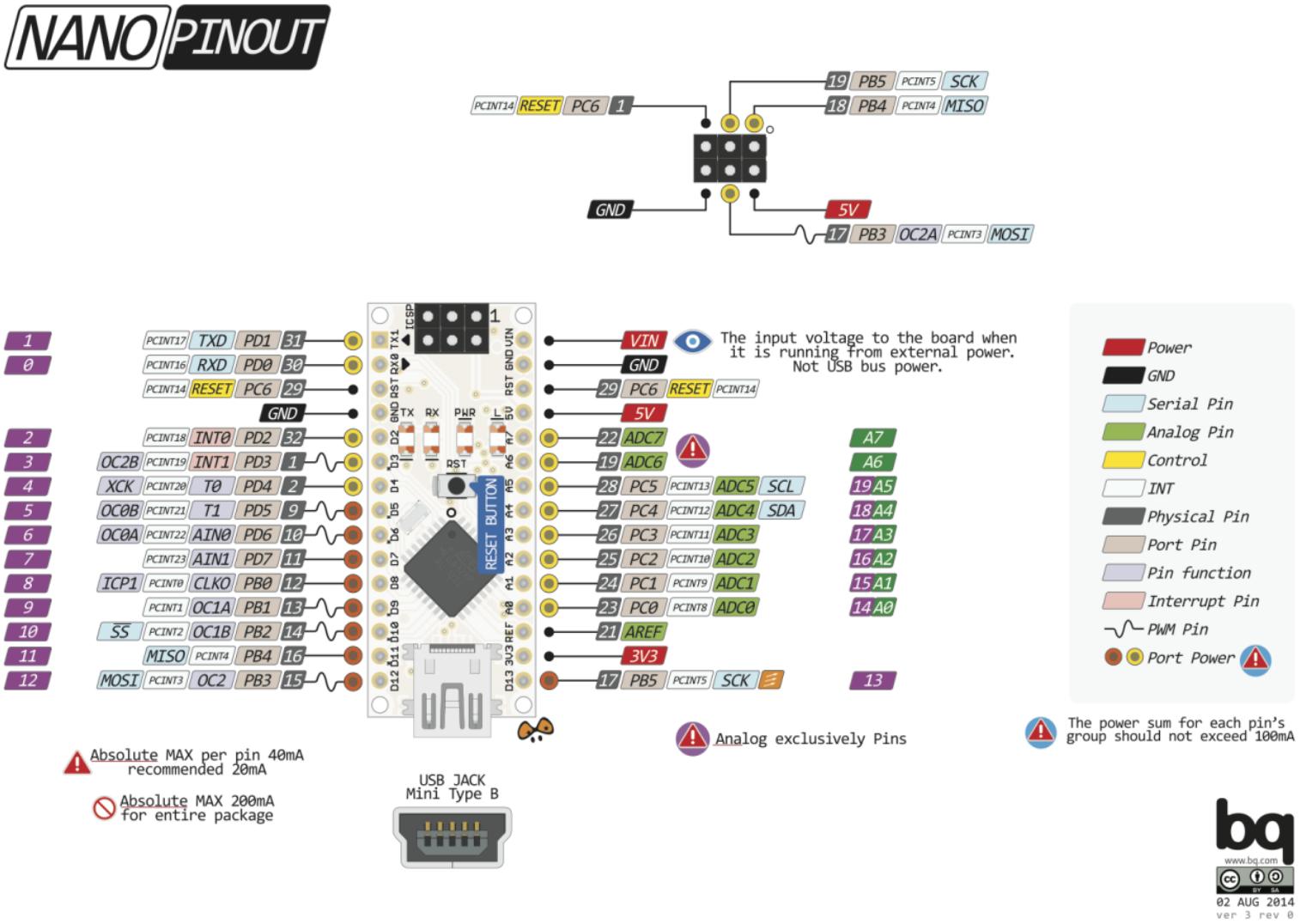
## RECOMENDACIONES

La solución mostrada en el presente trabajo es solamente un prototipo y no está diseñado para funcionar en ambientes reales. Una versión comercial para esta solución podría incluir el diseño de los PCB para los circuitos involucrados (en este proyecto han sido montados sobre Protoboards) y un adecuado montaje para los componentes y sus conexiones. Asimismo, como futuros trabajos se pueden mejorar ciertos aspectos del diseño, con la finalidad de crear un sistema más completo, con más funcionalidades y con mayor escalabilidad. Dentro de las recomendaciones para la mejora se pueden considerar las siguientes:

- La aplicación web es de libre acceso para cualquier usuario, lo que podría ocasionar una inadecuada gestión del control de la vivienda por parte de lo mismos, así como también dejar inoperativo los servicios debido al ataque de hackers. Para superar este inconveniente se recomienda diseñar un sistema de inicio de sesión para la plataforma web, con el cual únicamente puedan acceder a la plataforma los usuarios que estén registrados y autorizados en la misma. Implementar esta funcionalidad requiere de conocimientos avanzados en desarrollo web y deben ser desarrollada por un especialista, por esta razón no fue incluida en el diseño actual.
- Las conexiones entre el Arduino NANO y el Raspberry Pi son cableadas (comunicación serial), un deterioro en los medios físicos de estas conexiones podría quitarle funcionalidad al sistema. Ante este hecho se recomienda reemplazar las conexiones cableadas entre los Arduino UNO y el Raspberry Pi por conexiones inalámbricas del tipo Wifi, mediante el uso de módulos externos. La única desventaja de esta solución es que elevaría el costo de la plataforma.
- El sistema que controla la cocina no considera el hecho de un posible fallo al momento de realizar el encendido de la misma, es decir, cuando el sistema de ignición de la cocina no logre encender correctamente la hornilla y como consecuencia el gas podría seguir fluyendo. Esto podría ocasionar accidentes en las viviendas. Se recomienda para futuras implementaciones adicionar en el diseño un sistema de protección para este tipo de fallas.

# ANEXOS

# ANEXO A DESCRIPCIÓN DE PINES PARA EL ARDUINO NANO



## ANEXO B PROGRAMA EN PYTHON PARA EL CONTROLADOR CENTRAL

**Nombre del archivo:** *controlador_central.py*

```
import serial
import time
import mysql.connector

mydb =
mysql.connector.connect(host="51.158.77.11",user="pruebas",passwd="Vict
or1234",port='3306', database="tesis")

cur = mydb.cursor()
cur.execute("UPDATE dispositivos SET comand='0', estado='0' WHERE 1")
mydb.commit()
mydb.close()

while True:
            try:
            puerto1 = serial.Serial('/dev/ttyUSB0',9600,timeout=1)
            puerto2 = serial.Serial('/dev/ttyUSB1',9600,timeout=1)
            puerto3 = serial.Serial('/dev/ttyUSB2',9600,timeout=1)
            break
            except serial.SerialException:
            print "Faltan conectar alguno de los dispositivos"
            time.sleep(5)

tini = 0
tfin = 0

time.sleep(1.8)

while True:
            mydb =
mysql.connector.connect(host="51.158.77.11",user="pruebas",passwd="Vict
or1234",port='3306',database="tesis")
            cur = mydb.cursor()
            cur.execute("SELECT comand, estado FROM dispositivos WHERE
1")
            res = cur.fetchall()

            print res

######## Logica HERVIDOR ########
            ah = int(res[1][0])
            bh = int(res[1][1])

            ttrans = tfin - tini
            if ah==1 and bh ==0:
            puerto2.write('1')
            #print "PRENDIENDO"
            cur.execute("UPDATE dispositivos SET comand='0',estado='1'
WHERE id_disp='hervidor'" )
            mydb.commit()
            tini = time.time()
            elif ah==2 and bh==1:
            puerto2.write('2')
            #print "APAGANDO"
            cur.execute("UPDATE dispositivos SET comand='0', estado='0'
WHERE id_disp='hervidor'")
```

```
            mydb.commit()
            tini = 0
            tfin = 0
            elif bh==1:
            tfin = time.time()

            if ttrans > 600.0:
            cur.execute("UPDATE dispositivos SET comand='0', estado='0'
WHERE id_disp='hervidor'")
            mydb.commit()
            tini = 0
            tfin = 0

######## Logica COCINA ########
            ac = int(res[0][0])
            bc = int(res[0][1])

            if ac==1 and bc==0:
            puerto3.write('1')
            cur.execute("UPDATE dispositivos SET comand='0',estado='1'
WHERE id_disp='cocina1'")
            mydb.commit()
            elif ac==2 and bc==1:
            puerto3.write('2')
            cur.execute("UPDATE dispositivos SET comand='0',estado='0'
WHERE id_disp='cocina1'")
            mydb.commit()

######## Logica Luces Cocina ########
            alc = int(res[2][0])
            blc = int(res[2][1])

            if alc==1 and blc==0:
            puerto1.write('1')
            cur.execute("UPDATE dispositivos SET comand='0',estado='1'
WHERE id_disp='lcocina'")
            mydb.commit()
            elif alc==2 and blc==1:
            puerto1.write('2')
            cur.execute("UPDATE dispositivos SET comand='0',estado='0'
WHERE id_disp='lcocina'")
            mydb.commit()

######## Logica Luces Sala ########
            als = int(res[4][0])
            bls = int(res[4][1])

            if als==1 and bls==0:
            puerto1.write('3')
            cur.execute("UPDATE dispositivos SET comand='0',estado='1'
WHERE id_disp='lsala'")
            mydb.commit()
            elif als==2 and bls==1:
            puerto1.write('4')
            cur.execute("UPDATE dispositivos SET comand='0',estado='0'
WHERE id_disp='lsala'")
            mydb.commit()

######## Logica Luces Dormitorio ########
```

```
            ald = int(res[3][0])
            bld = int(res[3][1])

            if ald==1 and bld==0:
            puerto1.write('5')
            cur.execute("UPDATE dispositivos SET comand='0',estado='1'
WHERE id_disp='lcuarto'")
            mydb.commit()
            elif ald==2 and bld==1:
            puerto1.write('6')
            cur.execute("UPDATE dispositivos SET comand='0',estado='0'
WHERE id_disp='lcuarto'")
            mydb.commit()

######## CERRANDO CONEXION ########
            mydb.close()
            time.sleep(0.2)


puerto1.close()
puerto2.close()
puerto3.close()
```

## ANEXO C PROGRAMAS EN HTML PARA LA INTERFAZ WEB

**Nombre del archivo:** index.html

```
<!DOCTYPE html>

<html>
<head>
<meta charset="utf-8">
<title>Proyecto de Tesis</title>
<link rel="stylesheet" href="./css/style.css">
<link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/flexboxgrid/6.3.1/flexboxg
rid.min.css">
<link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/animate.css/3.5.2/animate.
min.css">
<link rel="shortcut icon" href="images/logo.png" type="image/png"/>
</head>
<body>
<div class="row main-container middle-xs center-xs">
<div class="col-md-8 col-sm-10 col-xs-11 col-lg-7">
<div class="box">
<div class="card body animated fadeInUp">
<header class="main-header">
<nav class="main-nav">
<a href="ambientes.html" class="nav-link">Explorar</a>
<a href="acerca.html" class="nav-link">Aviso</a>
</nav>
</header>
<article class="animated fadeInLeft delay-1">
<header class="text-center">
<img src="./images/logo.png" height="200px" alt="Logo UNI">
<h1 class="red-text threeD">Control Remoto de Hogares</h1>
</header>
</article>
</div>
</div>
</div>
</div>
</body>
</html>
```

**Nombre del archivo:** *acerca.html*

```
<!DOCTYPE html>

<html>
<head>
<meta charset="utf-8">
<title>Proyecto de Tesis</title>
<link rel="stylesheet" href="./css/style.css">
<link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/flexboxgrid/6.3.1/flexboxg
rid.min.css">
<link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/animate.css/3.5.2/animate.
min.css">
<link rel="shortcut icon" href="images/logo.png" type="image/png"/>
</head>
```

```
<body>
<div class="row main-container middle-xs center-xs">
<div class="col-md-8 col-sm-10 col-xs-11 col-lg-7">
<div class="box">
<div class="card body animated fadeInUp">
<header class="main-header">
<nav class="main-nav">
<a href="index.html" class="nav-link">Inicio</a>
<a href="ambientes.html" class="nav-link">Explorar</a>
</nav>
</header>
<div class="">
<h1>Datos del Domicilio</h1>
<p>Dirección: Jiron Las Encinas 337, San Martin de Porres, Lima,
Perú</p>
<p>Ubicación en el Mapa:</p>
<iframe
src="https://www.google.com/maps/embed?pb=!1m18!1m12!1m3!1d975.60562804
20481!2d-77.05569071188359!3d-
12.0144067877370 84!2m3!1f0!2f0!3f0!3m2!1i1024!2i768!4f13.1!3m3!1m2!1s0x
0%3A0x0!2zMTLCsDAwJzUxLjkiUyA3N8KwMDMnMTguNSJX!5e0!3m2!1ses!2spe!4v1541
302282822" width="200" height="200" frameborder="0" style="border:0"
allowfullscreen></iframe><br><br>
<p>Desarrollado por Victor Tomanguilla Collazos. Todos los derechos
reservados.</p>
</div>
</div>
</div>
</div>
</div>
</body>
</html>
```

**Nombre del archivo:** ambientes.html

```
<!DOCTYPE html>

<html>
<head>
<meta charset="utf-8">
<title>Proyecto de Tesis</title>
<link rel="stylesheet" href="./css/style.css">
<link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/flexboxgrid/6.3.1/flexboxg
rid.min.css">
<link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/animate.css/3.5.2/animate.
min.css">
<link rel="shortcut icon" href="images/logo.png" type="image/png"/>
</head>
<body>
<div class="row main-container middle-xs center-xs">
<div class="col-md-8 col-sm-10 col-xs-11 col-lg-7">
<div class="box">
<div class="card body animated fadeInUp">
<header class="main-header">
<nav class="main-nav">
<a href="index.html" class="nav-link">Inicio</a>
```

```
<a href="acerca.html" class="nav-link">Aviso</a>
</nav>
</header>
<article>
<div class="menu body">
<header class="text-center">
<img src="./images/logo.png" height="200px" alt="Logo UNI">
<h1 class="red-text">Ambientes Disponibles</h1>
</header>
<div class="productos">
<div class="row">
<div class="col-xs-12 col-sm-6 col-md-4 product animated fadeInLeft
delay-1">
<a href="cocina.html" class="nav-amb">
<div class="card">
<img src="./images/kitchen.png">
<h3>Cocina</h3>
<p>3 Dispositivos</p>
</div>
</a>

</div>
<div class="col-xs-12 col-sm-6 col-md-4 product animated fadeInLeft
delay-2">
<a href="sala.html" class="nav-amb">
<div class="card">
<img src="./images/room.png">
<h3>Sala</h3>
<p>1 Dispositivo</p>
</div>
</a>
</div>
<div class="col-xs-12 col-sm-6 col-md-4 product animated fadeInLeft
delay-3">
<a href="dormitorio.html" class="nav-amb">
<div class="card">
<img src="./images/bed.png">
<h3>Dormitorio 1</h3>
<p>1 Dispositivo</p>
</div>
</a>
</div>
                                          </div>
</div>
</div>
</article>
</div>
</div>
</div>
</div>
</body>
</html>
```

**Nombre del archivo:** *cocina.html*

```
<!DOCTYPE html>

<html>
```

```
<head>
<meta charset="utf-8">
<title>Proyecto de Tesis</title>
<link rel="stylesheet" href="./css/style.css">
<link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/flexboxgrid/6.3.1/flexboxg
rid.min.css">
<link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/animate.css/3.5.2/animate.
min.css">
<link rel="shortcut icon" href="images/logo.png" type="image/png"/>
</head>
<body>
<div class="row main-container middle-xs center-xs">
<div class="col-md-8 col-sm-10 col-xs-11 col-lg-7">
<div class="box">
<div class="card body">
<header class="main-header">
<nav class="main-nav">
<a href="index.html" class="nav-link">Inicio</a>
<a href="ambientes.html" class="nav-link">Explorar</a>
</nav>
</header>
<article>
<div class="menu body">
<header class="text-center">
<img src="./images/kitchen.png" height="200px" alt="Logo UNI">
<h1 class="red-text">Ambiente Cocina</h1>
</header>
<div class="productos">
<div class="row">
<div class="col-xs-12 col-sm-6 col-md-4 product botof">      <div
class="card botof">
<img src="./images/stove.png">
<h3>Cocina</h3>
<form action="luces.php" method="post">
<input  type="hidden" name="ruta" value="cocina.html" />
<input type="hidden" name="dispo" value="cocina1" />
<input style="display:inline;" name="l_on" type="submit" value="ON" />
<input style="display:inline;" name="l_off" type="submit" value="OFF"
/>
</form><br><br>
</div>
                                        </div>
<div class="col-xs-12 col-sm-6 col-md-4 product">
<div class="card">
<img src="./images/water-heater.png">
<h3>Hervidor</h3>
<form action="luces.php" method="post">
<input type="hidden" name="ruta" value="cocina.html" />
<input type="hidden" name="dispo" value="hervidor" />
<input style="display:inline;" name="l_on" type="submit" value="ON" />
<input style="display:inline;" name="l_off" type="submit" value="OFF"
/>
</form><br><br>
                            </div>

</div>
<div class="col-xs-12 col-sm-6 col-md-4 product">
```

```
<div class="card">
<img src="./images/creativity.png">
<h3>Luces</h3>
<form action="luces.php" method="post">
<input type="hidden" name="ruta" value="cocina.html" />
<input type="hidden" name="dispo" value="lcocina" />
<input style="display:inline;" name="l_on" type="submit" value="ON" />
<input style="display:inline;" name="l_off" type="submit" value="OFF"
/>
</form><br><br>
                            </div>

</div>
                                            </div>
</div>
</div>
</article>
</div>
</div>
</div>
</div>
</body>
</html>
```

**Nombre del archivo:** *sala.html*

```
<!DOCTYPE html>

<html>
<head>
<meta charset="utf-8">
<title>Proyecto de Tesis</title>
<link rel="stylesheet" href="./css/style.css">
<link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/flexboxgrid/6.3.1/flexboxg
rid.min.css">
<link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/animate.css/3.5.2/animate.
min.css">
<link rel="shortcut icon" href="images/logo.png" type="image/png"/>
</head>
<body>
<div class="row main-container middle-xs center-xs">
<div class="col-md-8 col-sm-10 col-xs-11 col-lg-7">
<div class="box">
<div class="card body"><!-- animated fadeInUp">-->
<header class="main-header">
<nav class="main-nav">
<a href="index.html" class="nav-link">Inicio</a>
<a href="ambientes.html" class="nav-link">Explorar</a>
</nav>
</header>
<article>
<div class="menu body">
<header class="text-center">
<img src="./images/room.png" height="200px" alt="Logo UNI">
<h1 class="red-text">Ambiente Sala</h1>
</header>
```

```
<div class="productos">
<div class="row">
<div class="col-xs-12 col-sm-6 col-md-4 product"><!-- animated
fadeInLeft delay-1">-->
<div class="card">
<img src="./images/creativity.png">
<h3>Luces</h3>
<form action="luces.php" method="post">
<input type="hidden" name="ruta" value="sala.html" />
<input type="hidden" name="dispo" value="lsala" />
<input style="display:inline;" name="l_on" type="submit" value="ON" />
<input style="display:inline;" name="l_off" type="submit" value="OFF"
/>
</form>

</div>
                                        </div>

               </div>
</div>
</div>
</article>
</div>
</div>
</div>
</div>
</body>
</html>
```

**Nombre del archivo:** *dormitorio.html*

```
<!DOCTYPE html>

<html>
<head>
<meta charset="utf-8">
<title>Proyecto de Tesis</title>
<link rel="stylesheet" href="./css/style.css">
<link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/flexboxgrid/6.3.1/flexboxg
rid.min.css">
<link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/animate.css/3.5.2/animate.
min.css">
<link rel="shortcut icon" href="images/logo.png" type="image/png"/>
</head>
<body>
<div class="row main-container middle-xs center-xs">
<div class="col-md-8 col-sm-10 col-xs-11 col-lg-7">
<div class="box">
<div class="card body"><!-- animated fadeInUp">-->
<header class="main-header">
<nav class="main-nav">
<a href="index.html" class="nav-link">Inicio</a>
<a href="ambientes.html" class="nav-link">Explorar</a>
</nav>
</header>
<article>
```

```
<div class="menu body">
<header class="text-center">
<img src="./images/bed.png" height="200px" alt="Logo UNI">
<h1 class="red-text">Ambiente Dormitorio 1</h1>
</header>
<div class="productos">
<div class="row">
<div class="col-xs-12 col-sm-6 col-md-4 product"><!-- animated
fadeInLeft delay-1">-->
<div class="card">
<img src="./images/creativity.png">
<h3>Luces</h3>
<form action="luces.php" method="post">
<input type="hidden" name="ruta" value="dormitorio.html" />
<input type="hidden" name="dispo" value="lcuarto" />
<input style="display:inline;" name="l_on" type="submit" value="ON" />
<input style="display:inline;" name="l_off" type="submit" value="OFF"
/>
</form>

</div>
                                        </div>

</div>
</div>
</div>
</article>
</div>
</div>
</div>
</div>
</body>
</html>
```

**Nombre del archivo:** *luces.php*

```
<?php
$servern = 'localhost';
$usern = "root";
$contra = "Elsa1234";
$dbname = "tesis";
$ruta = $_POST['ruta'];
$dispo = $_POST['dispo'];

$conn = new mysqli($servern,$usern,$contra,$dbname);

if (!conn){
    die("Fallo la conexion".mysqli_connect_error());
}

if(isset($_REQUEST['l_on'])){
    $valor = "1";
}else if(isset($_REQUEST['l_off'])){
    $valor = "2";
}

$sqla = "UPDATE dispositivos SET comand='$valor' WHERE
id_disp='$dispo'";
```

```
if($conn->query($sqla)){
    echo "Actualizado exitosamente";
}else{
    echo "Error actualizando".$conn->error;
}

$conn->close();
header("Location: http://51.158.77.11/$ruta");
?>
```

**Nombre del archivo:** *style.css*

```
body{
            font-family:Arial;
}

.body{
            padding: 60px 15px;
}

.main-header{
            background-color:rgba(113, 22, 16, 0.89);
            padding-left:15px;
            padding-right:15px;
            padding-top:10px;
            padding-bottom:10px;
            border-radius:4px;
}

.nav-link{
            color:white;
            width:49%;
            display:inline-block;
            text-decoration:none;
}

.nav-amb{
            text-decoration:none;
            color:black;
}

.text-center{
            text-align:center;
}

.red-text{
            color:#E74C3C;
}

.main-container{
            min-height:100vh;
}

.product img{
            width:100px;
}
```

```
.product h3{
            font-size:1.7em;
            margin-top:5px;
            margin-bottom:5px;
}

.botof{
            margin-bottom:15px;
}
.card{
            border-radius:4px;
            background-color:white;
            box-shadow:0 1px 3px rgba(0,0,0,0.12), 0 1px 2px
rgba(0,0,0,0.24);
}

.threeD{
            text-shadow: 1px 1px 0px #6b251d, 1px 5px 2px
rgb(171,168,168);
}

.delay-1{
            animation-delay: 0.2s;
            position: relative;
            z-index:5;
}

.delay-2{
            animation-delay: 0.4s;
            position: relative;
            z-index:4;
}

.delay-3{
            animation-delay: 0.6s;
            position: relative;
            z-index:3;
}

.delay-4{
            animation-delay: 0.8s;
            position: relative;
            z-index:2;
}

.delay-5{
            animation-delay: 1s;
            position: relative;
            z-index:1;
}
```

## ANEXO D FORMULARIO PARA LA ENCUESTA DE VALIDACIÓN

# Control Remoto de Hogares

Todos los datos obtenidos en la realización de esta encuesta serán usados únicamente con fines académicos. Se garantiza la privacidad de la información que pueda ser proporcionada al completar esta encuesta.

* Required

Nombre Completo: *

Your answer

Edad: *

Your answer

¿Alguna vez a escuchado hablar del término Domótica? *

- SI
- NO

¿Estaría interesado en un producto que le permita controlar los dispositivos de su hogar desde cualquier parte del mundo? *

- SI
- NO

¿Qué tan fácil de usar le parece la siguiente interfaz? *



- Muy Fácil
- Fácil
- Ni Fácil ni Dificil
- Dificil
- Muy Dificil

SUBMIT

Never submit passwords through Google Forms.

## BIBLIOGRAFÍA


1. Balboa J., Página web de la Empresa Panorama BPO, “Edificios inteligentes en Perú”. [En Línea]. Disponible en: http://www.panoramabpo.com/edificios-inteligentes-en-peru/. [Accedido: 22 octubre, 2018]
2. Sub Gerencia de Imagen Corporativa Banco de la Nación, “El Banco de la Nación inauguró su nueva sede, el edificio más alto del país”, 2016. [En línea]. Disponible en: www.bn.com.pe/noticias/2016/08072016-nota-prensa-ignauracion-nueva-sede.html. [Accedido: 22 octubre, 2018]
3. Versión Digital Diario El Comercio, “Así se verá el nuevo Centro de Convenciones de Lima”, 2014. [En Línea]. Disponible en: https://elcomercio.pe/lima/vera-nuevo-centro-convenciones-lima-310305. [Accedido: 22 octubre, 2018]
4. Versión Digital Diario Perú21, “Nueva sede de Biblioteca Nacional será una de las más modernas de Latinoamérica”, 2006. [En Línea]. Disponible en: http://archivo.peru21.pe/noticia/75794/nueva-sede-biblioteca-nacional-mas-modernas-latinoamerica. [Accedido: 22 octubre, 2018]
5. Página web de UTEC, “UTEC presenta el campus universitario considerado el más moderno de Latinoamérica”. [En Línea]. Disponible en: https://www.utec.edu.pe/noticias/utec-presenta-el-campus-universitario-considerado-el-mas-moderno-de-latinoamerica. [Accedido: 22 octubre, 2018]
6. Página web de la Empresa Smart House Perú, “Descubra el Confort y la Seguridad de Vivir en una Casa Inteligente”. [En Línea]. Disponible en: http://www.smarthouseperu.com/. [Accedido: 22 octubre, 2018]
7. Página web de la Empresa Activa Building Control, “Activa Ambientes Inteligentes”. [En Línea]. Disponible en: http://www.e-activa.com/. [Accedido: 22 octubre, 2018]
8. Página web de la Empresa S.R. Engineering Solutions S.A.C., “Domótica LCN, La Solución Segura e Innovadora para su Hogar, Oficina e Industria”. [En Línea]. Disponible en: http://www.lcnperu.com/. [Accedido: 22 octubre, 2018]
9. Cheng D., “Desarrollo e implementación de aplicaciones domóticas controladas con dispositivo Android”, tesis de grado, Universidad Nacional de Ingeniería, Lima, Perú, 2014.
10. Paiz J., “Diseño e implementación de una conexión remota bidireccional usando teléfono con sistema operativo Android para terminales de potencia aplicados en domótica”, tesis de grado, Universidad Nacional de Ingeniería, Lima, Perú,2015.
11. Página web de la Empresa Domintell, “¿Cuánto cuesta la instalación domótica de una vivienda?”. [En Línea]. Disponible en: http://www.domintell.es/presupuesto. [Accedido: 22 octubre, 2018]

12. Bouhai N., Saleh I., “Internet of Things: Evolutions and Innovations”, Gran Bretaña: Wiley, 2017, pp. 5-8.

13. Página web de Investopedia, “Wearable Technology”. [En Línea]. Disponible en: https://www.investopedia.com/terms/w/wearable-technology.asp. [Accedido: 22 octubre, 2018]

14. Página web de Sofos, “Internet de las Cosas”, 2015. [En Línea]. Disponible en: http://www.sofoscorp.com/internet-de-las-cosas/. [Accedido: 22 octubre, 2018]

15. Página web de Hipertextual, “¿Qué es y cómo funciona el Internet de las cosas?”, 2014. [En Línea]. Disponible en: https://hipertextual.com/archivo/2014/10/internet-cosas/. [Accedido: 22 octubre, 2018]

16. Página web del Protocolo CoAP, “RFC 7252 Constrained Application Protocol”. [En Línea]. Disponible en: http://coap.technology/. [Accedido: 22 octubre, 2018]

17. Página web del Protocolo MQTT, “MQTT.org”. [En Línea]. Disponible en: http://mqtt.org/. [Accedido: 22 octubre, 2018]

18. Página web de la Asociación Española de Domótica e Inmótica, “Qué es Domótica”. [En Línea]. Disponible en: http://www.cedom.es/sobre-domotica/que-es-domotica. [Accedido: 27 octubre, 2018]

19. Página web del Estándar KNX, “What is KNX?”. [En Línea]. Disponible en: https://www2.knx.org/za/knx/association/what-is-knx/index.php. [Accedido: 27 octubre, 2018]

20. Página web de la Empresa LonMark International, “What is the LonWorks Platform?”. [En Línea]. Disponible en: https://www.lonmark.org/connection/what_is_lon. [Accedido: 27 octubre, 2018]

21. Heath S., "Embedded Systems Design", Segunda Edición, Oxford: Newnes, 2002, pp. 2-14.

22. Página web de la Open Source Initiative, “About the Open Source Initiative". [En Línea]. Disponible en: https://opensource.org/about. [Accedido: 27 octubre, 2018]

23. Página web de Arduino, “What is Arduino?”. [En Línea]. Disponible en: https://www.arduino.cc/en/Guide/Introduction. [Accedido: 27 octubre, 2018]

24. Página web de Symmetry Electronics, “Arduino Development Kits: Advantages and Features”, 2014. [En Línea]. Disponible en: https://www.semiconductorstore.com/blog/2014/Arduino-Development-Kits-Advantages-and-Features/811/. [Accedido: 10 noviembre, 2018]

25. Página web de Arduino, “Arduino NANO”. [En Línea]. Disponible en: https://store.arduino.cc/usa/arduino-nano. [Accedido: 10 noviembre, 2018]

26. Página web de Arduino, “Arduino NANO (v2.3) User Manual”. [En Línea]. Disponible en: https://www.arduino.cc/en/uploads/Main/ArduinoNanoManual23.pdf. [Accedido: 10 noviembre, 2018]

27. Página web de Arduino, “Arduino Software (IDE)”. [En Línea]. Disponible en: https://www.arduino.cc/en/Main/Software. [Accedido: 10 noviembre, 2018]

28. Página web de Arduino, “Language Reference”. [En Línea]. Disponible en: https://www.arduino.cc/reference/en/. [Accedido: 11 noviembre, 2018]

29. Página web de Raspberry Pi, “Raspberry Pi, A small and affordable computer that you can use to learn programming”. [En Línea]. Disponible en: https://www.raspberrypi.org/. [Accedido: 27 octubre, 2018]

30. Página web de Raspberry Pi, “Raspbian”. [En Línea]. Disponible en: https://www.raspberrypi.org/downloads/raspbian/. [Accedido: 17 noviembre, 2018]

31. Página web de BalenEtcher, “Flash. Flawless.”. [En Línea]. Disponible en: https://www.balena.io/etcher/. [Accedido: 17 noviembre, 2018]

32. Página web Raspberry Forums, “Start Pixel desktop from console?”, 2016. [En Línea]. Disponible en: https://www.raspberrypi.org/forums/viewtopic.php?t=165536. [Accedido: 17 noviembre, 2018]

33. Página Web de Make Use Of, “7 Tips for Using a Raspberry Pi 3 as a Desktop PC with Raspbian”, 2018. [En Línea]. Disponible en: https://www.makeuseof.com/tag/raspberry-pi-3-desktop-pc-raspbian/. [Accedido 17 noviembre, 2018]

34. El Blog de Adrián, “Configura tu Raspberry Pi desde cero”, 2013. [En Línea]. Disponible en: https://adrianhontoria.wordpress.com/2013/11/18/configura-tu-raspberry-pi-desde-cero/. [Accedido: 17 noviembre, 2018]

35. Página web de CCM, “Comandos de Linux”, 2017. [En Línea]. Disponible en: https://es.ccm.net/contents/311-comandos-de-linux. [Accedido: 17 noviembre, 2018]

36. Página web de Hostinger, “¿Cómo funciona el SSH?”, 2017. [En Línea]. Disponible en: https://www.hostinger.es/tutoriales/que-es-ssh. [Accedido: 17, noviembre, 2018]

37. Página web de Genbeta, “¿Qué son y para qué sirven los hash?: funciones de resumen y firmas digitales”, 2013. [En Línea]. Disponible en: https://www.genbeta.com/desarrollo/que-son-y-para-que-sirven-los-hash-funciones-de-resumen-y-firmas-digitales. [Accedido: 17 noviembre, 2018]

38. Página web de PuTTY, “Download PuTTY”. [En Línea]. Disponible en: https://www.putty.org/. [Accedido: 17 noviembre, 2018]

39. Página web de UbuntuHandbook, “How to Enable SSH in Ubuntu 16.04 LTS”, 2016. [En Línea]. Disponible en: http://ubuntuhandbook.org/index.php/2016/04/enable-ssh-ubuntu-16-04-lts/. [Accedido: 17 noviembre, 2018]

40. Página web de Ekiketa, “Habilitar conexión ssh en Raspberry Pi”, 2016. [En Línea]. Disponible en: https://ekiketa.es/habilitar-conexion-ssh-en-raspberry-pi/. [Accedido: 17 noviembre, 2018]

41. Bahit E., “Curso: Python para Principiantes”, Argentina, 2012.

42. Cortés M., “Curso Moderno de Máquinas Eléctricas Rotativas”, Tomo II, Barcelona: Editores Técnicos Asociados, 1995, pp. 151-189.

43. Enríquez G., “El ABC del Control Electrónico de las Máquinas Eléctricas”, México: Limusa, 2003, pp. 324-327.

44. Página web de Aprendiendo Arduino, “Uso de Motores con Arduino”. [En Línea]. Disponible en: https://aprendiendoarduino.wordpress.com/tag/motor-dc/. [Accedido: 23 noviembre, 2018]

45. Página web de Texas Instruments, “L293x Quadruple Half-H Drivers”, 2016. [En Línea]. Disponible en: http://www.ti.com/lit/ds/symlink/l293.pdf. [Accedido: 23 noviembre, 2018]

46. Graf R., “Diccionario Moderno de Electrónica”, México: Prentice Hall Hispanoamericana, 1999, pp. 597-598

47. Rapetti G., Página web de Inventables, “Introducción a los Relés”. [En Línea]. Disponible en: https://www.inventable.eu/introduccion-a-los-reles/. [Accedido: 23 noviembre, 2018]

48. Página web de Hetpro, “Tarjeta interfaz de potencia con relevador 1CH”. [En Línea]. Disponible en: https://hetpro-store.com/tarjeta-interfaz-de-potencia-con-relevador-1ch/. [Accedido: 23 noviembre, 2018]

49. Ruiz C., Página web de OpenLanuza, “Utilizando un relé en Arduino”, 2018. [En Línea]. Disponible en: http://openlanuza.com/utilizando-un-rele-en-arduino/. [Accedido: 23 noviembre, 2018]

50. Página web de Ecured, “Servidor Web”. [En Línea]. Disponible en: https://www.ecured.cu/Servidor_Web. [Accedido: 24 noviembre, 2018]

51. Bearnes B., Página web de DigitalOcean, “¿Cómo instalar Linux, Apache, MySQL, PHP (LAMP) en Ubuntu 16.04?”, 2016. [En Línea]. Disponible en: https://www.digitalocean.com/community/tutorials/como-instalar-linux-apache-mysql-php-lamp-en-ubuntu-16-04-es. [Accedido: 24 noviembre, 2018]

52. Pérez D., Página web de Maestros del Web, “¿Qué son las bases de datos?”, 2007. [En Línea]. Disponible en: http://www.maestrosdelweb.com/que-son-las-bases-de-datos/. [Accedido: 24 noviembre, 2018]

53. Bearnes B., Página web de DigitalOcean, “¿Cómo Instalar y Proteger phpMyAdmin en Ubuntu 16.04?”, 2016. [En Línea]. Disponible en:

https://www.digitalocean.com/community/tutorials/como-instalar-y-proteger-phpmyadmin-en-ubuntu-16-04-es. [Accedido: 24 noviembre, 2018]

54. Página web de Android, “Android, The world’s most popular mobile OS”. [En Línea]. Disponible en: https://www.android.com/. [Accedido: 26 noviembre, 2018]

55. Página web de ScaleWay, “Scaleway: Enjoy a new cloud experience”. [En Línea]. Disponible en: https://www.scaleway.com/. [Accedido: 26 noviembre, 2018]

56. Página web de ScaleWay, “How to create and enable SSH Keys”. [En Línea]. Disponible en: https://www.scaleway.com/docs/configure-new-ssh-key/. [Accedido: 26 noviembre, 2018]

57. Página web de ScaleWay, “How to create and connect to your first instance”. [En Línea]. Disponible en: https://www.scaleway.com/docs/create-and-connect-to-your-server/. [Accedido: 26 noviembre, 2018]

58. Sverdlov E., Página web de DigitalOcean, “Crear un nuevo usuario y otorgarle permisos en MySQL”, 2014. [En Línea]. Disponible en: https://www.digitalocean.com/community/tutorials/crear-un-nuevo-usuario-y-otorgarle-permisos-en-mysql-es. [Accedido: 26 noviembre, 2018]

59. Página web de Freenom, “freenom, A Name for Everyone”. [En Línea]. Disponible en: https://www.freenom.com/. [Accedido: 26 noviembre, 2018]

60. Página web de Raspberry Pi, “Installing Python Packages”. [En Línea]. Disponible en: https://www.raspberrypi.org/documentation/linux/software/python.md. [Accedido: 26 noviembre, 2018]

61. Página web de Pynative, “Install MySQL Connector Python on Windows, MacOs, Linux, Unix and Ubuntu”, 2018. [En Línea]. Disponible en: https://pynative.com/install-mysql-connector-python/. [Accedido: 26 noviembre, 2018]

62. Hernández U., Página web de CódigoFacilito, “Mi primera página web”. [En Línea]. Disponible en: https://codigofacilito.com/cursos/crear-pagina-web. [Accedido 26, noviembre, 2018]

63. Página web de MIT App Inventor, “Anyone Can Build Apps That Impact the World”. [En Línea]. Disponible en: http://appinventor.mit.edu/. [Accedido: 26 noviembre, 2018]

64. Página web de Google Forms, “Google Forms: Free Online Surveys for Personal Use”. [En Línea]. Disponible en: https://www.google.com/forms/about/.. [Accedido: 27 noviembre, 2018]

65. Página Web de Mercado Libre Perú, “Mercado Libre”. [En Línea]. Disponible en: https://www.mercadolibre.com.pe/. [Accedido: 27 noviembre, 2018]