



# **Tarjeta Smart House para una habitación**

**César Andres Tejada Torres  
John Alejandro Barahona Pineda**

Universidad del Quindío  
Facultad de Ingeniería, Ingeniería Electrónica  
Armenia, Colombia  
2018



# **Tarjeta Smart House para una habitación**

**César Andres Tejada Torres  
John Alejandro Barahona Pineda**

Tesis o trabajo de grado presentada(o) como requisito parcial para optar al título de:  
**Ingeniero Electrónico**

Director:  
Ing. César Augusto Álvarez Gaspar

Línea de Investigación:  
Internet de las Cosas  
Universidad del Quindío  
Facultad de Ingeniería, Ingeniería Electrónica  
Armenia, Colombia  
2018



# **Agradecimientos**

**John Alejandro Barahona Pineda**

**César Andres Tejada Torres**



## Resumen

El ón.

**Palabras clave:** (máximo 10 palabras, preferiblemente seleccionadas de las listas internacionales que permitan el indizado cruzado).

## Abstract

Endarín.

**Keywords:** palabras clave en inglés(máximo 10 palabras, preferiblemente seleccionadas de las listas internacionales que permitan el indizado cruzado)

# Contenido

<b>Agradecimientos</b>	v
<b>Resumen</b>	VII
<b>1 Objetivos</b>	x
1.1 Objetivo General . . . . .	X
1.2 Objetivos Específicos . . . . .	X
<b>Lista de símbolos</b>	xi
<b>2 Introducción</b>	1
<b>3 Marco Teórico</b>	2
3.1 Internet de las Cosas . . . . .	2
3.1.1 Plataforma Heroku . . . . .	2
3.1.2 Framework Laravel . . . . .	3
3.1.3 HTTP . . . . .	3
3.1.4 JSON . . . . .	4
3.1.5 Base de Datos . . . . .	5
3.2 Smart House . . . . .	5
3.3 Hardware . . . . .	5
3.3.1 ESP-WROOM-32 . . . . .	5
3.3.2 Corriente Alterna (AC) . . . . .	6
3.3.3 Corriente Directa (DC) . . . . .	7
3.3.4 Control de potencia AC por ángulo de fase . . . . .	7
3.3.5 Control de Cargas DC . . . . .	8
3.3.6 I2C . . . . .	8
3.3.7 Sensores . . . . .	8
3.4 Software . . . . .	13
3.4.1 RTOS . . . . .	13
3.4.2 ESP-IDF . . . . .	14
3.4.3 Proteus . . . . .	15
<b>4 Desarrollo e Implementación</b>	16
4.1 Hardware . . . . .	16

4.2	Firmware . . . . .	16
4.3	Software . . . . .	17
<b>5</b>	<b>Resultados y Análisis</b>	<b>20</b>
5.1	Software . . . . .	20
5.1.1	Parte Pública . . . . .	21
5.1.2	Parte Privada . . . . .	22
5.1.3	Base de Datos . . . . .	24
5.2	Firmware . . . . .	24
5.2.1	Conexión a Internet vía Wi-Fi . . . . .	24
5.2.2	Escritura de Datos en la Aplicación Web . . . . .	25
5.2.3	Lectura de Datos de Internet . . . . .	25
<b>6</b>	<b>Conclusiones</b>	<b>26</b>
<b>7</b>	<b>Trabajos Futuros</b>	<b>27</b>
<b>8</b>	<b>Glosario</b>	<b>28</b>
	<b>Bibliografía</b>	<b>30</b>

# **1 Objetivos**

## **1.1. Objetivo General**

Desarrollar una solución IoT para una habitación en un entorno de Smart House.

## **1.2. Objetivos Específicos**

- Desarrollar un prototipo de una tarjeta inalámbrica para una habitación en un entorno Smart House.
- Desarrollar una aplicación web encargada de permitir la interacción del usuario con su habitación.
- Evaluar el desempeño del sistema en una prueba beta.

# **Lista de símbolos**

## **Abreviaturas**

<b>Abreviatura</b>	<b>Término</b>
<i>RF</i>	Radiofrecuencia
<i>IR</i>	Infrarrojo
<i>SMS</i>	Servicio de Mensajes Cortos
<i>GSM</i>	Sistema Global para las comunicaciones Móviles
<i>SBC</i>	Computadoras de Placa Simple
<i>IoT</i>	Internet de las Cosas
<i>AC</i>	Corriente alterna
<i>DC</i>	Corriente continua
<i>PaaS</i>	Plataforma como Servicio
<i>JSON</i>	Notación de Objetos de JavaScript

## **2 Introducción**

Con el avance de las tecnologias, y el crecimiento exponencial que ha tenido el internet, es posible enlazar comunicaciones con infinidad de dispositivos en multiples lugares del mundo, accediendo a todo tipo de informacion, desde series de television, hasta datos reales generados por procesos industriales. Para que esto sea posible, se han creado sistemas capaces de enviar datos a la red (internet), con multiples fines, ya sea para estudios estadisticos o simple informacion general. Uno de los objetivos principales de IoT, es manejar esta informacion de manera inteligente, es decir, el proceso de tomar desiciones y acciones significativas con esta informacion.

IoT puede ser implementado tanto a gran escala, como a una ciudad, o a una escala mas reducida, como por ejemplo una habitacion, sea una cocina, un pequena bodega de alimentos, etc., en la cual existen multiples variables a tener en cuenta. Para el analisis de las multiples variables presentes en una habitacion, se requiere un sistema capaz de, capturar la informacion, tomar desiciones inteligentes sobre esta informacion, para posteriormente ejecutar acciones que puedan influir directa o indirectamente con dichas variables.

La gran ventaja de IoT en relacion a la toma de desiciones inteligentes sobre la informacion, es la capacidad de darle a esa informacion un proposito mayor, ya que puede ser empleado para algun tipo de realimentacion con fines de mejorar procesos o como se menciono anteriormente, con propósitos generales.

# **3 Marco Teórico**

## **3.1. Internet de las Cosas**

La internet de las cosas es un sistema de dispositivos de computación interrelacionados, máquinas mecánicas y digitales, objetos, animales o personas que tienen identificadores únicos y la capacidad de transferir datos a través de una red, sin requerir de interacciones humano a humano o humano a computadora.

IoT ha evolucionado desde la convergencia de tecnologías inalámbricas, sistemas microelectromecánicos, microservicios e Internet. La convergencia ha ayudado a derribar las paredes de silos entre la tecnología operativa y la tecnología de la información, permitiendo que los datos no estructurados generados por máquinas sean analizados para obtener información que impulse mejoras. [1]

Kevin Ashton, cofundador y director ejecutivo del Auto-ID Center de MIT, mencionó por primera vez la internet de las cosas en una presentación que hizo a Procter & Gamble en 1999. He aquí cómo Ashton explica el potencial de la internet de las cosas:

“Las computadoras de hoy –y, por lo tanto, la internet– dependen casi totalmente de los seres humanos para obtener información. Casi todos los aproximadamente 50 petabytes (un petabyte son 1.024 terabytes) de datos disponibles en internet fueron capturados y creados por seres humanos escribiendo, presionando un botón de grabación, tomando una imagen digital o escaneando un código de barras.

El problema es que la gente tiene tiempo, atención y precisión limitados, lo que significa que no son muy buenos para capturar datos sobre cosas en el mundo real. Si tuviéramos computadoras que supieran todo lo que hay que saber acerca de las cosas –utilizando datos que recopilaron sin ninguna ayuda de nosotros– podríamos rastrear y contar todo, y reducir en gran medida los desechos, las pérdidas y el costo. Sabríamos cuándo necesitamos reemplazar, reparar o recordar cosas, y si eran frescas o ya pasadas”. [2]

### **3.1.1. Plataforma Heroku**

“Heroku es una plataforma en la nube basada en un sistema gestionado por contenedores, con servicios de datos integrados y un potente ecosistema, para desarrollar y ejecutar aplicaciones.

ciones modernas. La experiencia de los desarrolladores de Heroku es un enfoque centrado en aplicaciones para la entrega de software, integrado con las herramientas y flujos de trabajo de desarrollador más populares de la actualidad” [3].

### 3.1.2. Framework Laravel

“Laravel es un framework de aplicaciones web con una sintaxis expresiva y elegante. Laravel intenta aliviar el dolor del desarrollo al facilitar las tareas comunes que se utilizan en la mayoría de los proyectos web, como la autenticación, el enrutamiento, las sesiones y el almacenamiento en caché.

Laravel tiene como objetivo hacer que el proceso de desarrollo sea agradable para el desarrollador sin sacrificar la funcionalidad de la aplicación. Con este fin, se ha intentado combinar lo mejor de lo que hemos visto en otros marcos web, incluidos los marcos implementados en otros lenguajes, como Ruby on Rails, ASP.NET MVC y Sinatra.

Laravel es accesible, pero potente, y proporciona potentes herramientas necesarias para aplicaciones grandes y robustas. Una magnífica inversión de contenedores de control, un sistema de migración expresivo y un soporte de prueba de unidades estrechamente integrado le brindan las herramientas que necesita para construir cualquier aplicación” [4].

### 3.1.3. HTTP

El protocolo de transferencia de hipertexto (HTTP) es un protocolo de la capa de aplicación en el modelo OSI, se usa para transmitir documentos hipermedia, como HTML. Fue diseñado para la comunicación entre navegadores web y servidores web, pero también se puede usar para otros propósitos. HTTP sigue un modelo clásico de cliente-servidor, con un cliente que abre una conexión para realizar una petición, luego esperando la respuesta. HTTP es un protocolo sin estado, significa que el servidor no conserva ningún dato entre dos solicitudes. Para realizar las peticiones este protocolo cuenta con diferentes métodos [5].

**Metodos de solicitud HTTP:** HTTP define un conjunto de métodos de solicitudes para indicar la acción deseada que se realizará para un recurso determinado. Aunque pueden ser sustantivos, estos métodos algunas veces se denominan verbos HTTP. Cada uno de ellos implementa una semántica diferente, pero algunas características comunes son compartidas por un grupo de ellos [6].

- GET: este método solicita una representación del recurso especificado. Las peticiones que lo usan, solo deben regresar datos.

- HEAD: este método solicita una respuesta igual a una petición GET, pero sin el cuerpo de la respuesta.
- POST: se usa para enviar una entidad al recurso especificado, causando a menudo un cambio en el estado o efectos secundarios en el servidor.
- PUT: reemplaza todas las representaciones actuales del recurso destino con la la petición de carga útil.
- DELETE: esta petición elimina el recurso especificado.
- CONNECT: establece un túnel para el servidor identificado por el recurso de destino.
- OPTIONS: se usa para describir las opciones de la comunicación para el recurso destino.
- TRACE: realiza una prueba de mensaje loop-back a lo largo de la ruta del recurso de destino.
- PATCH: se usa para realizar modificaciones parciales a un recurso.

### 3.1.4. JSON

“JSON (JavaScript Object Notation - Notación de Objetos de JavaScript) es un formato ligero de intercambio de datos. Leerlo y escribirlo es simple para humanos, mientras que para las máquinas es simple interpretarlo y generarlo. Está basado en un subconjunto del Lenguaje de Programación JavaScript, Standard ECMA-262 3rd Edition - Diciembre 1999. JSON es un formato de texto que es completamente independiente del lenguaje pero utiliza convenciones que son ampliamente conocidos por los programadores de la familia de lenguajes C, incluyendo C, C++, C#, Java, JavaScript, Perl, Python, y muchos otros. Estas propiedades hacen que JSON sea un lenguaje ideal para el intercambio de datos” [7].

JSON está constituido por dos estructuras:

- Una colección de pares de nombre/valor. En varios lenguajes esto es conocido como un objeto, registro, estructura, diccionario, tabla hash, lista de claves o un arreglo asociativo.
- Una lista ordenada de valores. En la mayoría de los lenguajes, esto se implementa como arreglos, vectores, listas o secuencias.

Estas son estructuras universales; virtualmente todos los lenguajes de programación las soportan de una forma u otra. Es razonable que un formato de intercambio de datos que es independiente del lenguaje de programación se base en estas estructuras [7].

### 3.1.5. Base de Datos

“Se define una base de datos como una serie de datos organizados y relacionados entre sí, los cuales son recolectados y explotados por los sistemas de información de una empresa o negocio en particular.

Cada base de datos se compone de una o más tablas que guarda un conjunto de datos. Cada tabla tiene una o más columnas y filas. Las columnas guardan una parte de la información sobre cada elemento que queramos guardar en la tabla, cada fila de la tabla conforma un registro” [8]. Es muy común encontrar en la gestión de las bases de datos por parte del usuario las funciones básicas de crear, leer, actualizar y borrar (CRUD), estas se crean directamente en una interfaz para que el usuario pueda interactuar con la base de datos.

## 3.2. Smart House

El concepto de Smart House implica tres características básicas. En primer lugar, el monitoreo a través de redes de sensores para obtener información sobre la casa y sus residentes. En segundo lugar, los mecanismos que controlan el uso de la comunicación entre dispositivos para permitir la automatización y el acceso remoto. Por último, las interfaces de usuario, como los teléfonos inteligentes y las computadoras que permiten a los usuarios especificar las preferencias, así como presentar información a las personas acerca de estas preferencias.

Smart House es un entorno que tiene sistemas sofisticados a través de los cuales se pueden controlar algunas de las cosas de la casa, como luces, puertas, ventanas, además puede racionalizar el consumo de energía, entre otras funciones mediante el uso de sensores. Básicamente, uno de los beneficios más importantes del uso de la tecnología en las casas, es la prestación de servicios a las personas.[9]

## 3.3. Hardware

### 3.3.1. ESP-WROOM-32

Es un potente módulo MCU Wi-Fi + BT + BLE que se dirige a una amplia variedad de aplicaciones, desde redes de sensores de baja potencia hasta las tareas más exigentes, como codificación de voz, transmisión de música y decodificación de MP3, además de su reducido tamaño, como se observa en la figura 3-1.

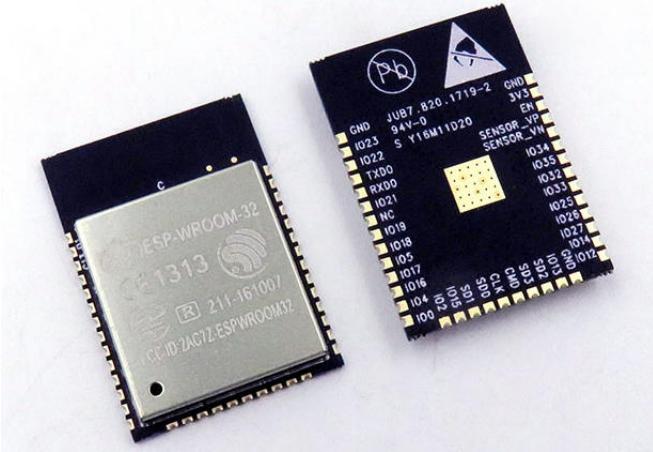
En el núcleo de este módulo está el chip ESP32-D0WDQ6. El chip integrado está diseñado para ser escalable y adaptable. Hay dos núcleos de CPU que se pueden controlar individualmente, y la frecuencia del reloj es ajustable de 80 MHz a 240 MHz. El usuario también puede apagar la CPU y utilizar el coprocesador de baja potencia para monitorear constantemente

los periféricos en busca de cambios o cruces de umbrales. ESP32 integra un amplio conjunto de periféricos, que van desde sensores táctiles capacitivos, sensores Hall, interfaz de tarjeta SD, Ethernet, SPI de alta velocidad, UART, I2S e I2C.

La integración de Bluetooth, Bluetooth LE y Wi-Fi garantiza que se pueda orientar una amplia gama de aplicaciones, el uso de Wi-Fi permite un gran alcance físico y conexión directa a Internet a través de Wi-Fi, mientras usa Bluetooth, le permite al usuario conectarse convenientemente al teléfono o transmitir balizas de baja energía para su detección. La corriente de reposo del chip ESP32 es inferior a 5 uA, lo que lo hace adecuado para aplicaciones de electrónica con batería y portátiles. ESP32 admite una velocidad de datos de hasta 150 Mbps y una potencia de salida de 20.5 dBm en la antena para garantizar el rango físico más amplio.

El sistema operativo elegido para ESP32 es freeRTOS con LwIP; TLS 1.2 con aceleración de hardware está integrado también. También se admite la actualización segura (cifrada) a través del aire (OTA), de modo que los desarrolladores puedan actualizar continuamente sus productos incluso después de su lanzamiento.[10]

**Figura 3-1:** ESP WROOM 32[11]



### **3.3.2. Corriente Alterna (AC)**

“Es un tipo de corriente eléctrica, en la que la dirección del flujo de electrones va y viene a intervalos regulares o en ciclos. La corriente que fluye por las líneas eléctricas y la electricidad disponible normalmente en las casas procedente de los enchufes de la pared es corriente alterna. La corriente estándar utilizada en los EE.UU. es de 60 ciclos por segundo (es decir, una frecuencia de 60 Hz); en Europa y en la mayor parte del mundo es de 50 ciclos por segundo (es decir, una frecuencia de 50 Hz.)”. [12]

### 3.3.3. Corriente Directa (DC)

“Es la corriente eléctrica que fluye de forma constante en una dirección, como la que fluye en una linterna o en cualquier otro aparato con baterías es corriente continua.

Una de las ventajas de la corriente alterna es su relativamente económico cambio de voltaje. Además, la pérdida inevitable de energía al transportar la corriente a largas distancias es mucho menor que con la corriente continua”. [12]

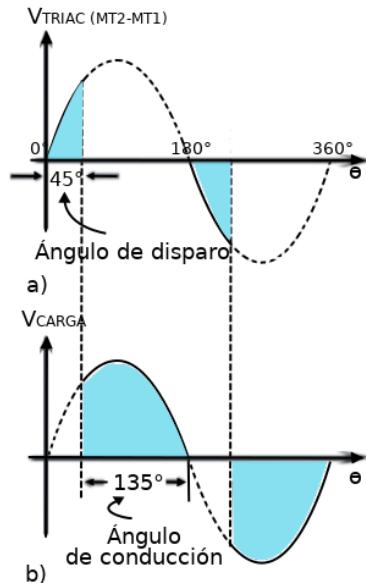
### 3.3.4. Control de potencia AC por ángulo de fase

“Los SCR y los TRIAC, permiten aplicar una técnica muy conveniente y eficaz para controlar el voltaje promedio y por lo tanto la potencia aplicada a una carga, cambiando el ángulo de fase con el cual la fuente de voltaje se aplica a ésta. Esta técnica de control de voltaje es muy usada en las aplicaciones de regulación de motores, iluminación y temperatura, por ser el voltaje la variable principal en estos tres procesos”.[13]

“Para entender como se controla el ángulo de fase, por medio de un TRIAC conectado en serie con la carga, se puede asumir que el TRIAC se comporta idealmente como un interruptor controlador por la corriente de compuerta Ig que se cierra o se abre ante su presencia o ausencia. Observando la figura 3-2 puede verse el control de una onda seno de tensión con un período con un período de 360 grados; en la parte (a) de la figura se muestra la tensión a través del TRIAC, mientras que en la parte (b) se ve la tensión sobre la carga; allí puede verse que el TRIAC se comporta como un circuito abierto durante los primeros 45 grados de cada semiciclo, y todo el voltaje cae en sus terminales eliminando el flujo de corriente por la carga. La porción del semiciclo durante la cual se presenta esta situación se conoce como ángulo de disparo”[13].

“Una vez el TRIAC es disparado a través de su terminal de compuerta (G), éste se engancha y se comporta como un interruptor cerrado, permitiendo que todo el voltaje se aplique a la carga durante los 135 grados restantes de cada semiciclo. La porción del semiciclo durante la cual el TRIAC conduce se denomina ángulo de conducción”[13].

**Figura 3-2:** Representación gráfica del ángulo de disparo y de conducción del TRIAC y de la carga [13].



### 3.3.5. Control de Cargas DC

Las cargas de corriente continua típicas como los motores y LED's, a parte de poder funcionar en dos estados, encendido y apagado, pueden ser controladas mediante la modulación por ancho de pulso (PWM), ya que esta

### 3.3.6. I2C

El protocolo

### 3.3.7. Sensores

#### Módulo GY-30

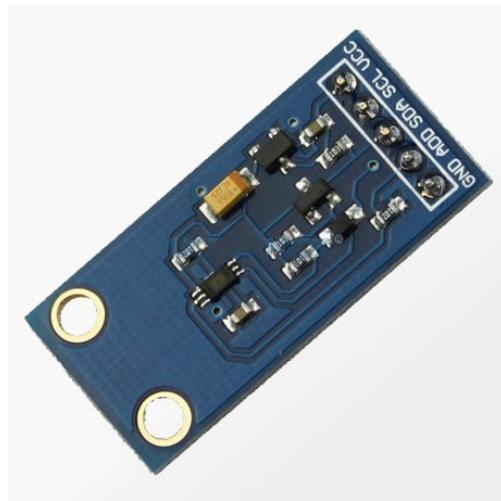
“Sensor GY-30 BH1750FVI. Es un sensor digital de intensidad de luz ambiente, tiene un conversor ADC de 16bits interno y comunicación por I2C como se observa en la figura 3-3. Esta es una versión mejorada del típico sensor de luz a base de un LDR, el cual simplemente entrega un valor analógico. Compatible con Arduino, PIC, etc.”

El módulo BH1750 es un sensor de luz, que a diferencia del LDR es digital y nos entrega valores de medición en Lux ( lumen /m<sup>2</sup> ) que es una unidad de medida estándar para el nivel de iluminación (iluminancia). Tiene alta precisión y un rango entre 1 – 65535 lx el cual

es configurable.

La interfaz de comunicación es I2C pudiéndolo implementar en la mayoría de micro controladores, el módulo aparte de los pines de alimentación y pines I2C tiene un pin para establecer la dirección”.[14]

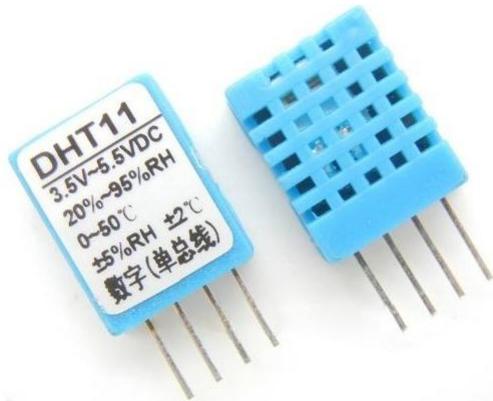
**Figura 3-3:** Modulo GY-30 [14]



### Temperatura y Humedad DHT11

“El DHT11 es un sensor de temperatura y humedad digital de bajo costo. Utiliza un sensor capacitivo de humedad y un termistor para medir el aire circundante, y muestra los datos mediante una señal digital en el pin de datos (no hay pines de entrada analógica). Es bastante simple de usar, pero requiere sincronización cuidadosa para tomar datos. El único inconveniente de este sensor es que sólo se puede obtener nuevos datos una vez cada 2 segundos, así que las lecturas que se pueden realizar serán mínimo cada 2 segundos”. [15]

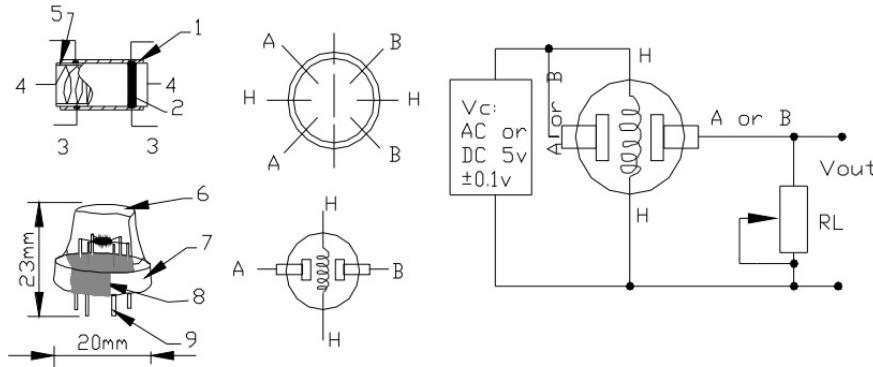
**Figura 3-4:** Sensor de temperatura y humedad DHT11 [15]



### Módulo sensor de calidad de aire MQ-135

La serie MQ de sensores de gas son sensores analógicos por lo que son fáciles de implementar con cualquier microcontrolador que posea un conversor analógico digital (ADC) adecuado. Estos sensores son electroquímicos y varán su resistencia con la exposición a determinados gases, internamente poseen un calentador que se encarga de aumentar la temperatura interna para que el sensor pueda reaccionar con los gases provocando un cambio de valor en la resistencia, su estructura interna se puede observar en la figura 3-5.[16]

**Figura 3-5:** Estructura del sensor MQ [16]



El Sensor Calidad Aire MQ135 se utilizan en equipos de control de calidad del aire para edificios y oficinas, son adecuados para la detección de NH<sub>3</sub>, NOx, alcohol, benceno, humo, CO<sub>2</sub>, etc. Además de que estos sensores vienen en módulos como se observa en la figura 3-6, lo que facilita el uso de estos, simplemente se debe conectar al microcontrolador sin necesidad de hacer algún circuito de acople. [16]

**Figura 3-6:** Modulo sendor de calidad de Aire MQ-135 [16]



Este sensor es sensible en similar proporción a los gases mencionados, con lo que se puede

determinar si el aire está limpio o si existe presencia de algún gas nocivo.

Los valores de salida de este sensor no son valores absolutos, simplemente proporciona una salida analógica que se debe monitorear y ser comparada con los valores típicos proporcionados en la hoja de datos como se menciona anteriormente.[17]

### Sensores de Estado

Los sensores de estado describen si la variable está en alto (1) o en bajo (0), para estos sensores se tienen diferentes variables típicamente, como el estado de una puerta o una ventana (abierta o cerrada), la lluvia, el movimiento.

**Módulo detector de lluvia:** Es un módulo relativamente simple que consiste en una serie de pistas conductoras organizadas de forma paralela e impresas sobre una placa de baquelita como se observa en la figura 3-7. La separación entre sus pistas es muy pequeña, con el fin de crear un corto circuito cada vez que las pistas se mojan, ya que es un circuito abierto y el agua hace que se cree un camino de baja resistencia entre las pistas que tienen diferente potencial (Vcc-GND). La corriente que fluye a través de estas pistas se ve limitada por resistencias de 10K en cada conductor, lo que impide que el corto circuito que se genera cuando se moja la placa vaya a estropear el micro controlador.[18]

Figura 3-7: Sensor de Lluvia [18]

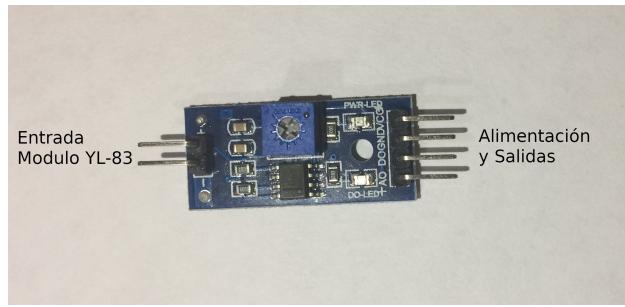


El circuito de control es el que posee las resistencias limitadoras de corriente y es el encargado de alimentar el módulo de la figura 3-7. Como se observa en la figura 3-8 posee un amplificador operacional, específicamente el circuito integrado LM392. Este es el encargado de amplificar el pequeño diferencial de voltaje que se genera cuando una gota de agua cae sobre las pistas del módulo. Aquí es donde se genera la señal de salida que puede ser del tipo

analógica o digital. La señal digital oscilará entre los valores HIGH y LOW dependiendo de si hay agua o no sobre las pistas de la placa.

La salida analógica entregará un nivel de voltaje que variará dependiendo de la cantidad de agua que haya sobre el módulo.[18]

**Figura 3-8:** Modulo de sensor de lluviar [Imagen Propia]

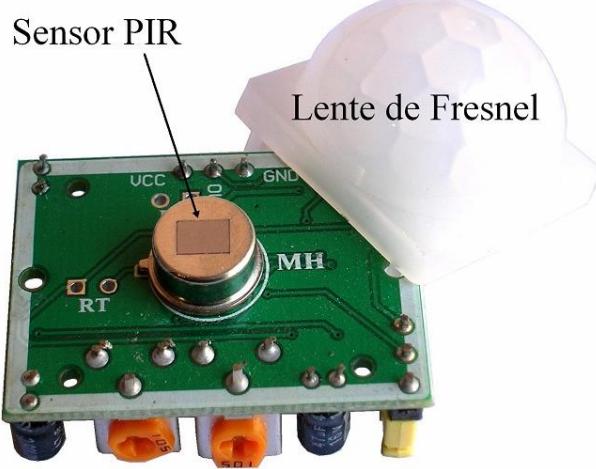


**Módulo PIR HC-SR501:** La función de los sensores PIR es detectar movimiento, normalmente se busca detectar el movimiento de una persona dentro del rango del sensor. Son baratos, pequeños, de bajo consumo y fáciles de utilizar, además no se desgastan. Normalmente se encuentran en electrodomésticos y gadgets para la oficina o el hogar. Son conocidos como PIR, “Sensores Infrarrojos” o “Sensores de movimiento”.

Este módulo contiene un sensor Piroeléctrico, el cual puede detectar niveles de radiación infrarroja como se observa en la figura 3-9. Este sensor de movimiento está dividido en 2 mitades, la razón para esto es que se busca la diferencia en el movimiento y no el promedio. Las dos mitades están unidas de modo que se cancelan una a otra. Entonces, si una mitad recibe más o menos radiación IR, la salida cambiará a Alto o Bajo. [19]

El módulo PIR modelo HC-SR501, es pequeño y de bajo costo como se observa en la figura , incorporando la tecnología más reciente en sensores infrarrojos pasivos para detectar movimiento. La emisión infrarroja se da en personas por su temperatura corporal y en animales (mamíferos), ya que emiten una radiación similar a los humanos. [20]

**Figura 3-9:** Sensor HCSR501 [20]



## 3.4. Software

### 3.4.1. RTOS

Los sistemas operativos en tiempo real, tienen como parámetro clave al tiempo, ya que en gran variedad de situaciones, como por ejemplo, un proceso industrial, se requiere recolectar múltiples datos, los cuales son usados para el control de múltiples procesos, los cuales deben ser ejecutados en determinados instantes, de no ser así, podría causar desde la mala ejecución de una tarea, hasta un accidente según la delicadeza del proceso.

Para procesos con nula tolerancia a fallos, se conoce como un sistema en tiempo real duro, muchos de estos sistemas se encuentran en el control de procesos industriales, en aeronáutica, en la milicia y en áreas de aplicación similares. El caso contrario, cuando se tiene cierta tolerancia a que muy ocasionalmente existan fallos, se conoce como sistema en tiempo real suave, los sistemas de audio digital o de multimedia están en esta categoría. Los teléfonos digitales también son ejemplos de sistema en tiempo real suave. [21]

“Como en los sistemas en tiempo real es crucial cumplir con tiempos predeterminados para realizar una acción, algunas veces el sistema operativo es simplemente una biblioteca enlazada con los programas de aplicación, en donde todo está acoplado en forma estrecha y no hay protección entre cada una de las partes del sistema. Un ejemplo de este tipo de sistema en tiempo real es freeRTOS. Las categorías de sistemas para computadoras de bolsillo, sistemas integrados y sistemas en tiempo real se traslanan en forma considerable. Casi todos ellos tienen por lo menos ciertos aspectos de tiempo real suave. Los sistemas integrados y de tiempo real sólo ejecutan software que colocan los diseñadores del sistema; los usuarios no

pueden agregar su propio software, lo cual facilita la protección.

Los sistemas de computadoras de bolsillo y los sistemas integrados están diseñados para los consumidores, mientras que los sistemas en tiempo real son más adecuados para el uso industrial. Sin embargo, tienen ciertas características en común". [21]

### 3.4.2. ESP-IDF

ESP-IDF es el entorno de desarrollo oficial para el ESP32 desarrollado por Espressif System, el cual mediante una serie de comandos específicos escritos en la terminal (para el caso de linux), permite realizar una configuración del ESP32 en cuanto a su funcionamiento, es decir, permite encender o apagar características como el WiFi, el Bluetooth o realizar particiones de memoria, ademas de esto, se puede cargar el código por el puerto USB al ESP32, al igual que se puede visualizar la información generada por el ESP32 por este mismo puerto. Este entorno se encuentra construido con diferentes características y APIs, algunas de ellas se mencionan a continuación. [22]

**FreeRTOS:** este framework esta desarrollado sobre este sistema operativo de tiempo real, tomando de este diferentes funcionalidades como la tareas y las colas.

**Wi-Fi:** el modulo ESP-WROOM-32 posee la capacidad en el hardware para la conexión a una red Wi-Fi, y este software proporciona las librerías y las funcionalidades para realizar dicha conexión y enviar o recibir datos.

**Particiones:** el MCU contiene una memoria flash, la cual por medio de este framework se le pueden realizar particiones de memoria, dedicadas a guardar el programa, y también para guardar ciertos archivos. Estas se configuran por medio de un fichero csv.

**Consola:** el framework provee una librería para realizar la programación de una consola, para realizar diferentes tareas dentro del sistema, las cuales se pueden programar.

**HTTP Request:** para realizar diferentes peticiones HTTP como las descritas anteriormente, el entorno de desarrollo cuenta con la librería LWIP para realizarlas.

**Timers:** este framework provee funciones para disponer de los timers de 64 bits que posee el esp32 y además para configurar sea un timer periódico o de un solo disparo.

### 3.4.3. Proteus

Proteus combina facilidad de uso con características de gran alcance para ayudar a diseñar, probar y el diseño de PCB profesionales. Con casi 800 variantes de microcontroladores listos para la simulación directamente desde el esquemático, además de contar con uno de los paquetes de diseño de PCB profesionales más intuitivas en el mercado y un autoruteo de clase mundial incluido como estándar. [23]

Además es una herramienta muy completa y potente de simulación de circuitos y diseño de PCBs. Dentro de la simulación de circuitos admite componentes pasivos, digitales, analógicos y componentes más complejos como LCDs y motores. Por tanto, se puede hacer casi cualquier cosa, el programa está pensado para que una vez se tenga el circuito diseñado se pueda pasar a una PCB [24]. Adicionalmente los dispositivos que no se encuentren se pueden agregar, ya sea de la página oficial o construyéndolos dentro de la herramienta.

# 4 Desarrollo

## 4.1. Hardware

El hardware se diseña por medio del software Proteus ...

## 4.2. Firmware

El firmware se desarrolla sobre el framework o SDK oficial de Espressif Systems, ESP-IDF el cual posee una documentación [22], la cual es muy útil a la hora de utilizar las diferentes APIs que este posee. Este firmware incluye un kernel de tiempo real llamado FreeRTOS, el cual da soporte al manejo de los diferentes recursos del sistema. Como es un RTOS, las funciones se definen mediante las tareas, entonces para cada funcionalidad de la tarjeta o grupo de funcionalidades se desarrolla una o varias tareas para que realicen las acciones adecuadas, por ejemplo, en cuanto a los sensores cada uno tiene una tarea para su lectura, gestión de estos datos, así, como también en cuanto a las diferentes salidas de la tarjeta se han agrupado en las cargas que solo son de encendido/apagado y las de control, creando diferentes tareas para esta gestión.

Sobre el firmware se desarrollan los siguientes temas:

**Tareas:** como se menciona anteriormente, estas cumplen las funciones principales en cuanto a toda la gestión de los diferentes periféricos y funcionalidades de la tarjeta, así que se desarrollan diferentes tareas para la lectura y escritura en los puertos de la tarjeta.

**GPIO:** el ESP-WROOM-32 posee diferentes GPIO, los cuales se usan para leer o escribir señales digitales, en cuanto a los sensores se pueden enviar señales para iniciar su lectura o simplemente tener el pin en modo entrada y leerlo cada cierto intervalo de tiempo para generar los datos de lectura, o en modo salida para el control de los diferentes dispositivos que se han desarrollado en el hardware.

**ADC,DAC:** los ADC se usan para leer los datos de algunos sensores que proporcionan datos analógicos, por este motivo se hace la conversión de la señal analógica a un valor digital dentro de la tarjeta, para luego identificar el valor de la lectura del sensor. Los DAC

se usan para realizar la operación contraria, teniendo valores digitales convertirlos a un valor analógico por ejemplo para generar audios o diferentes señales a partir del software.

**Consola:** para realizar diferentes pruebas directamente desde la tarjeta, se usa la opción de la consola que se comunica por medio del puerto serie, para esto se crean las funciones y los comandos que estaran disponibles. Algunos comandos disponibles son *http*, para realizar las peticiones http manualmente y observar su respuesta, *pin* para realizar la prueba de un pin digital como entrada o salida, *help* para observar la lista de comandos, entre otros.

**HTTP Request:** las peticiones HTTP son indispensables en estas aplicaciones del campo IOT, por este motivo en el desarrollo del firmware se usan las librerías pertinentes para realizar peticiones y además leer las respuestas de estas desde el servidor, ya que este es el medio de comunicación tarjeta-servidor.

**Hora de Red:** se obtiene la hora mediante el protocolo simple de tiempo de red (SNTP), este resulta de gran utilidad para la sincronización de los relojes de los sistemas informáticos.

**Timers:** la tarjeta posee dos grupos de timers, que cuenta cada uno con dos timers, un timer lo usa el sistema operativo, otro es configurado para realizar el control de potencia AC por ángulo de fase, para tener la sincronía necesaria con la señal de la red eléctrica.

**I2C:** el protocolo I2C se activa por medio de la instalación del driver en algún par de pines GPIO disponibles en la tarjeta. Se configura e inicia y posteriormente se crea una tarea la cual se encarga de solicitar y leer los datos de los diferentes sensores conectados a este.

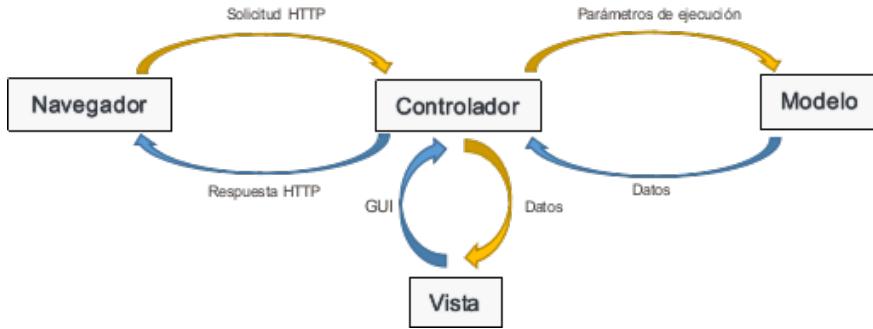
**PWM:** se ha mencionado anteriormente que para controlar las cargas DC se usa una salida PWM, el esp32 proporciona esta funcionalidad en algunos de sus pines, para su uso se configura y asignan los valores de funcionamiento.

**Interrupciones:** las interrupciones se usan para no gastar recursos en un monitoreo constante de las entradas, solo cuando existe un cambio de nivel en la entrada el dispositivo desencadena una serie de instrucciones relacionas al tipo de interrupción y a diferentes funciones creadas para esta, la interrupción se usa por medio de los diferentes pines propuestos para esto en el hardware.

## 4.3. Software

En esta sección se desarrolla una aplicación web, la cual se encarga de hacer la gestión entre el usuario y la tarjeta. De este modo, se usa un patron de arquitectura Modelo-Vista-Controlador (MVC) para esta aplicación, este modelo es realmente util ya que separa la

**Figura 4-1:** Modelo-Vista-Controlador

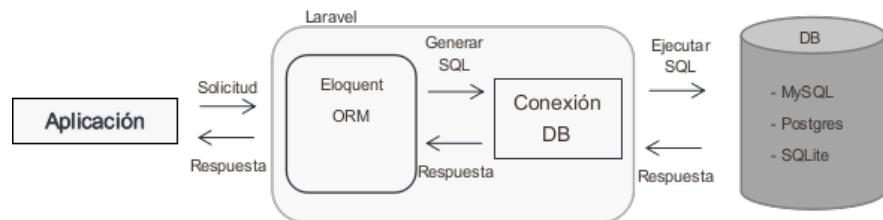


lógica de negocio de la interfaz de usuario, incrementando la reutilización y flexibilidad y además la escalabilidad de ambos aspectos por separado, dicho esto la aplicación cuenta con diferentes modelos, controladores y vistas. La función de cada parte de esta arquitectura se puede observar en la figura 4-1.

El flujo es el siguiente, primero el usuario realiza alguna acción en la interfaz (por ejemplo, presiona un botón, un enlace, etc), luego el controlador recibe (por parte de los objetos de la interfaz-vista) la notificación de la acción solicitada por el usuario. El controlador gestiona el evento que llega. Luego el controlador accede al modelo, actualizándolo, posiblemente modificándolo de forma adecuada a la acción solicitada por el usuario (por ejemplo, el controlador actualiza los datos del perfil del usuario) y después la interfaz de usuario espera nuevas interacciones del usuario, comenzando el ciclo nuevamente.

Este patrón de diseño se usa en la programación orientada a objetos, por lo tanto se realiza la aplicación en el lenguaje de programación PHP, ya que es realmente útil para realizar la gestión de peticiones y envíos de formularios en dicha aplicación, además de que es importante también la gestión de las bases de datos de la aplicación, por este motivo se utiliza un framework basado en este lenguaje y esta arquitectura, el cual realiza diferentes trabajos en cuanto a la parte de la arquitectura, para gestionar las diferentes partes de la aplicación, en este caso se usa el framework Laravel el cual como se menciona anteriormente esta orientado a facilitar las tareas comunes de la mayoría de proyectos web, que utilizan HTML5 y PHP.

Además, con este framework se hace uso de un ORM (Mapeo Objeto-Relacional) llamado Eloquent. Esta es una forma de mapear los datos que se encuentran en la base de datos a objetos de PHP y viceversa, esto facilita el uso de diferentes gestores de bases de datos como MySQL, SQLite, entre otras, ya que todas las consultas están en PHP y el ORM ya se encarga del mapeo a los comandos SQL como se observa en la figura 4-2. Eloquent usa los modelos para enviar y recibir la información de la base de datos.

**Figura 4-2:** ORM

# 5 Resultados y Análisis

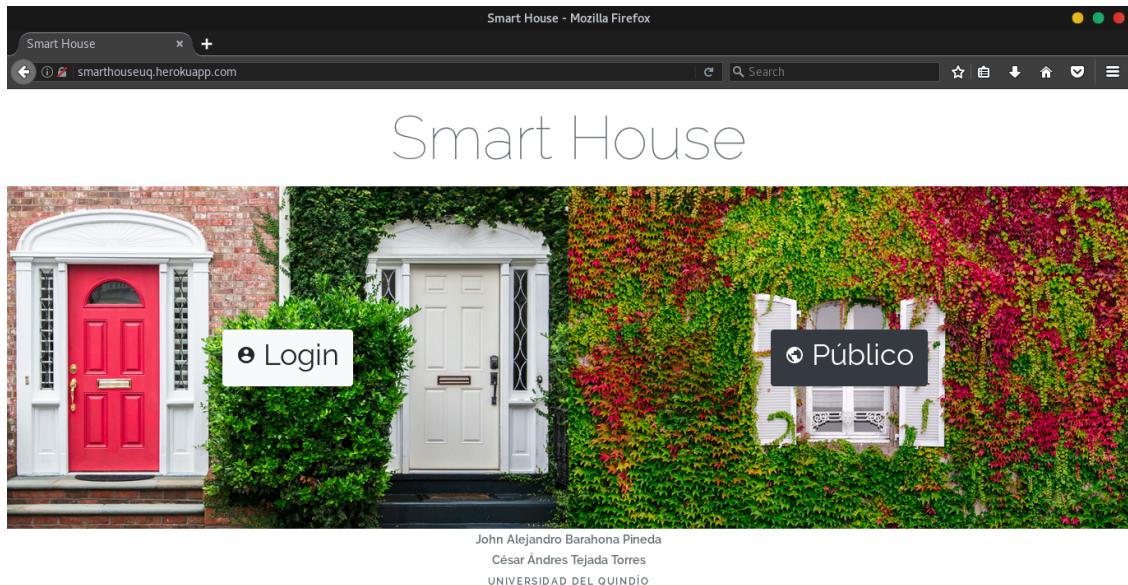
## 5.1. Software

Se desarrolla la aplicación web de manera local, y posteriormente se lanza a la web, esta compuesta por los siguientes sitios y las diferentes interacciones basadas en las funciones basicas, crear, leer, actualizar y borrar (CRUD).

- Parte Pública
  - API
  - Panel de Control
    - Crear
    - Ver
    - Editar
    - Eliminar

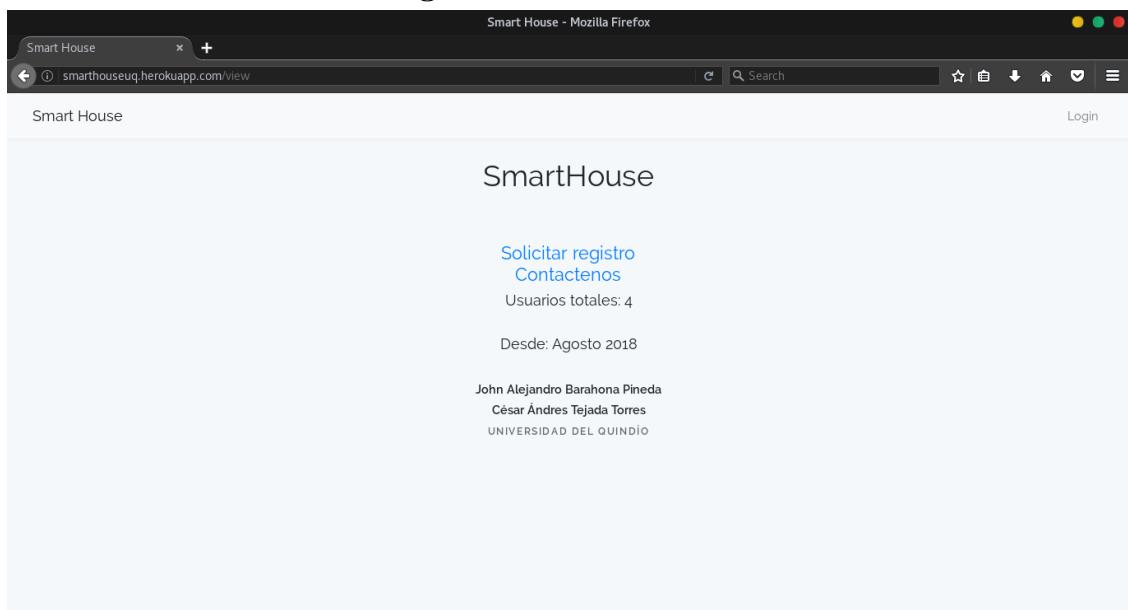
De acuerdo a la lista anterior, se toman en cuenta dos partes para esta, una pública y una privada, como se observa en la figura 5-1. En la parte pública se encuentra una vista con los datos de contacto, solicitudes de registro o productos y la cantidad de usuarios que actualmente estan registrados en la aplicación. En la parte privada se encuentra la interacción de los usuarios sea administrador, dueño de una casa o de una habitación, para controlar y ver sus datos.

Las diferentes interacciones que tiene cada usuario en el panel de control se garantizan por medio del framework, creando diferentes roles para cada usuario que se esta registrando y realizando la comprobación por parte de los controladores y el middleware que este provee.

**Figura 5-1:** Página de Inicio

### 5.1.1. Parte Pública

En esta vista únicamente hay opciones para el contacto y solicitudes, como se menciona anteriormente, es una vista sencilla dada la poca información que contiene, como se observa en la figura 5-2.

**Figura 5-2:** Vista Pública

### 5.1.2. Parte Privada

En esta sección es donde se encuentra el Panel de Control para los diferentes usuarios de la aplicación. En primera instancia, para un usuario administrador, que es el encargado de gestionar la aplicación, este usuario tiene la posibilidad de crear, ver, editar y eliminar los diferentes registros de la aplicación, la vista de este usuario se puede observar en la figura 5.4(a). Por medio de este usuario es que se activan las cuentas de los demás, por esto en la parte pública estás las opciones de contacto y solicitud de registro.

**Figura 5-3:** Vistas de Usuarios

Nombre	Apellido	Email	Rol	Acciones
Admin	Admin	admin@smarthouse.com	admin	
User	House	userhouse@smarthouse.com	userHouse	
User	Room2 H	userroom2h@smarthouse.com	userRoom	
User	Room	userroom@smarthouse.com	userRoom	
User	Room1 H	userroom1h@smarthouse.com	userRoom	

John Alejandro Barahona Pineda  
César Andrés Tejada Torres

(a) Usuario Administrador

Nombre	Apellido	Email	Rol	Acciones
User	Room2 H	userroom2h@smarthouse.com	userRoom	
User	Rooms H	userrooms@smarthouse.com	userRoom	

John Alejandro Barahona Pineda  
César Andrés Tejada Torres  
UNIVERSIDAD DEL QUINDIO

(b) Usuario de Casa

Nombre	Descripción	Casa	Estado
User Room1 H	UserRoom1 H	userroom1h@smarthouse.com	on

John Alejandro Barahona Pineda  
César Andrés Tejada Torres

(c) Usuario de Habitación

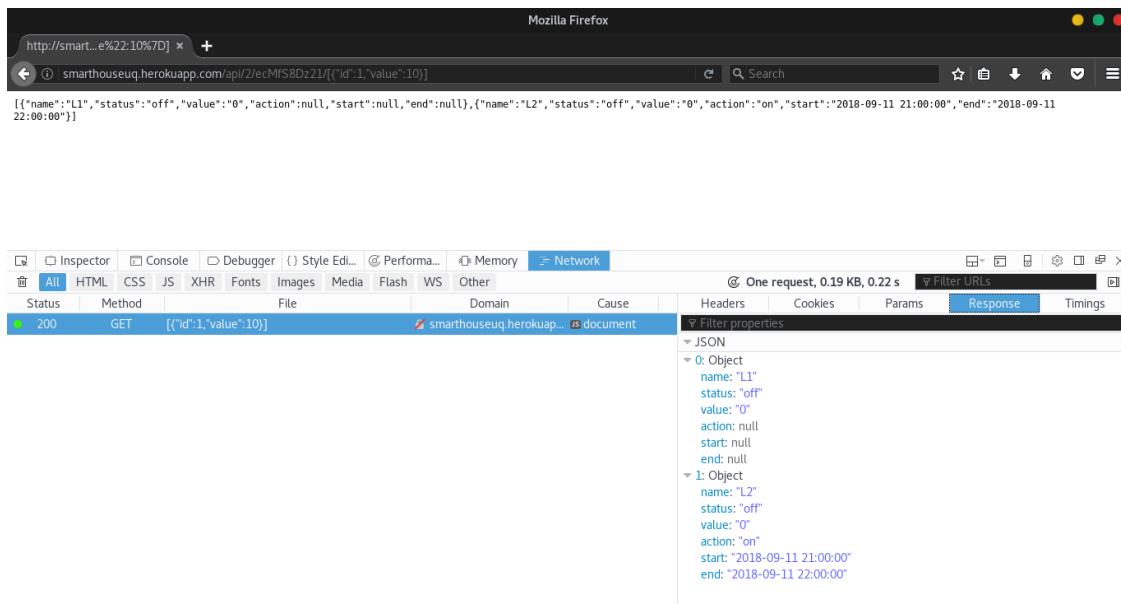
También existe el usuario dueño de la casa donde se encuentra el dispositivo, este usuario es opcional y es para gestionar los dispositivos presentes dentro de una misma casa, es un administrador de la casa, el cual puede ver y editar algunos campos de sus usuarios hijos o usuarios habitación y sus diferentes casas y habitaciones, únicamente las que estén regis-

tradas a su nombre, como se ve en la figura 5.4(b), de este modo el rol de este usuario es administrar su casa y visualizar los datos de esta.

Por último, otro rol es el de usuario habitación, el cuál es un usuario que solo visualiza sus propios datos, como la habitación y los dispositivos presentes en esta, como se observa en la figura 5.4(c), a este solo le compete la información de lo que posee en su habitación, por tal motivo el panel de control muestra una vista general de los datos y el estado de sus dispositivos, además de tener la capacidad de editar partes básicas de su habitación y perfil. Este usuario puede o no estar sujeto a un usuario padre o usuario casa, ya que, solamente puede poseer una tarjeta para su habitación y ninguna otra en dicha casa.

Además de este panel de control, desde el cuál se realizan las operaciones sobre la aplicación, en la parte privada se encuentra la ruta encargada de la actualización Servidor-Tarjeta, es decir, en esta ruta es donde se da la comunicación. En esta ruta se realiza una petición HTTP tipo GET por parte de la tarjeta, esta contiene el id de la habitación en la cuál esta instalada la tarjeta y también el token correspondiente a esta, y además en la URL se añade un texto tipo JSON en el cual se encuentra toda la información de la lectura actual de los sensores, esta petición la responde el servidor con un texto, también tipo JSON que contiene la información pertinente de las cargas o actuadores, como se observa la figura 5-4, para dar seguridad a esta transacción, se se utiliza el token mencionado anteriormente, el cual se verifica mediante el id de la habitación y que este coincida con los datos almacenados, de este modo se garantiza que lo que se envía este sometido a verificación.

**Figura 5-4:** Página de intercambio de datos



The screenshot shows a Mozilla Firefox browser window. The address bar displays the URL: `http://smart...e%22:10%7D]`. Below the address bar, the status bar shows the response: `[{"name": "L1", "status": "off", "value": "0", "action": null, "start": null, "end": null}, {"name": "L2", "status": "off", "value": "0", "action": "on", "start": "2018-09-11 21:00:00", "end": "2018-09-11 22:00:00"}]`.

The main content area of the browser shows the JSON response received from the server:

```
{
  "name": "L1",
  "status": "off",
  "value": "0",
  "action": null,
  "start": null,
  "end": null
},
{
  "name": "L2",
  "status": "off",
  "value": "0",
  "action": "on",
  "start": "2018-09-11 21:00:00",
  "end": "2018-09-11 22:00:00"
}
```

Below the browser window, the Firefox Network tab of the developer tools is visible. It shows a single request: One request, 0.19 KB, 0.22 s. The Response tab is selected, displaying the JSON data shown above. The Headers tab shows the following:

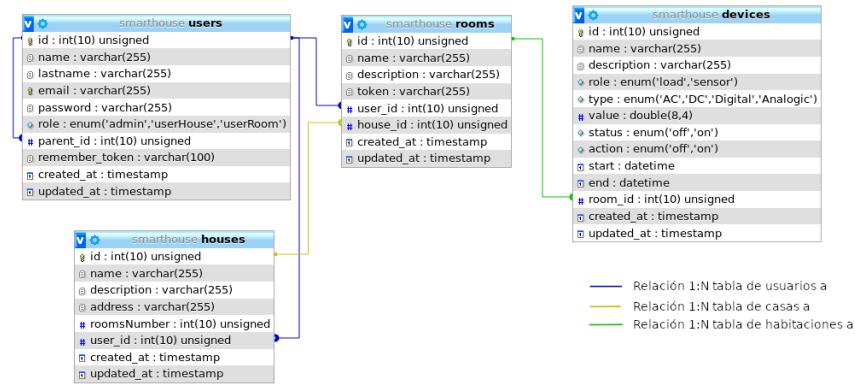
Status	Method	File	Domain	Cause	Headers	Cookies	Params	Response	Timings
200	GET	[{"id": 1, "value": 10}]	smarthouseug.herokuapp...	document					

The Response pane shows the JSON data again, with expanded objects for L1 and L2.

### 5.1.3. Base de Datos

La estructura de la base de datos se puede observar en la figura 5-5, aquí se observan los diferentes campos que posee cada tabla, además de las llaves y sus relaciones, las relaciones presentes en esta estructura son de tipo 1:N, es decir, por ejemplo un usuario puede tener relacionadas N casas.

**Figura 5-5:** Base de datos SmartHouse



## 5.2. Firmware

### 5.2.1. Conexión a Internet vía Wi-Fi

Los sistemas IoT deben estar conectados siempre a Internet, por este motivo se debe brindar un canal para decirle al sistema como acceder a este, por lo tanto, se desarrolla una tarea que se encarga de esto, como se ha mencionado el módulo del esp32 es capaz de funcionar como Punto de Acceso (AP) y como Cliente o Estación (STA) al mismo tiempo, aprovechando esta capacidad se usa un servidor local, el cual se encarga de gestionar la conexión de la tarjeta con Internet.

De este modo, como se observa en la figura XXX esta es la pagina principal del servidor local de la tarjeta, la cual permite conectarse a una red que este al alcance de la tarjeta, y también desconectarse de esta, su funcionamiento es muy intuitivo, se selecciona la red a la que se desea conectar la tarjeta, se ingresan sus credenciales y posteriormente el dispositivo verifica si la conexión fue exitosa o no, si la conexión es exitosa ya la tarjeta esta lista para su funcionamiento, basta con reiniciarla para que solo quede funcionando como STA y no en el modo dual, además de esto si las credenciales de la red cambian también se incluye un botón para el borrado de estas, de este modo se puede configurar de nuevo una red.

### **5.2.2. Escritura de Datos en la Aplicación Web**

Los datos que esta leyendo la tarjeta provienen de los diferentes sensores que tiene conectados como se ha mencionado, se usan sensores de estado para sensar la presencia, sensor de calidad de aire entre otros presentes en esta. Para la escritura de los datos, en el firmware, se desarrollan diferentes tareas encargadas de leer y enviar los datos a una tarea central, los datos que están enviando contienen el id del dispositivo y la medida que lee en ese momento, estos datos se envian en forma de texto en formato JSON, de este modo, la tarea central los gestiona y envía a la aplicación con el mismo formato, organizandolos en la petición HTTP tipo GET que realiza. La aplicación ya se encarga de almacenarlos y mostrarlos al usuario como se menciona anteriormente.

### **5.2.3. Lectura de Datos de Internet**

Al inicio de la aplicación, se sincroniza y se almacena la hora actual de la red, por medio del protocolo SNTP, esto con el propósito de poder gestionar las diferentes reglas solicitadas por el usuario, por ejemplo, que encienda o apague un dispositivo a una hora dada, después de obtener este dato la aplicación continua con normalidad para realizar las diferentes peticiones a la aplicación web.

La interacción del usuario se da con la aplicación web mencionada anteriormente, de este modo la tarjeta siempre se debe actualizar, para esto, cuando la tarjeta envía los datos de los sensores la aplicación responde con los datos necesarios para tarjeta, esta los recibe en una cadena texto en formato JSON, los procesa y los envía a las tareas pertinentes ya sea para encender o apagar algún dispositivo conectado a la tarjeta, además envía las reglas que el usuario ha definido.

## **6 Conclusiones**

## **7 Trabajos Futuros**

Realizar la implementación de una funcionalidad infrarroja para almacenar las diferentes señales de los controles remotos y poder controlar otros dispositivos.

Gestionar un servidor local de respaldo en la tarjeta para que este funcionando en caso de fallos de conexión a internet o con la aplicación web.

## 8 Glosario

**AC (Corriente alterna):** corriente eléctrica variable en la que las cargas eléctricas (electrones) cambian el sentido del movimiento a través de un conductor de manera periódica.

**DC (Corriente continua):** corriente de intensidad constante en la que el movimiento de las cargas eléctricas (electrones) siempre es en el mismo sentido.

**Internet del todo (IoE):** es un concepto que extiende el énfasis de la internet de las cosas (IoT) en las comunicaciones de máquina a máquina para describir un sistema más complejo que también abarca personas y procesos.[25]

**Internet de las Cosas (IoT):** parte fundamental del internet del todo (IdT), el cual se refiere principalmente a la interacción máquina-máquina, en incluso interacción máquina-persona.

**Software:** conjunto de programas y rutinas que permiten a un sistema realizar determinadas tareas.

**Hardware:** partes físicas que componen un sistema electrónico, como por ejemplo los componentes de un circuito electrónico.

**Firmware:** programa informático que establece la lógica de más bajo nivel que controla los circuitos electrónicos de un dispositivo de cualquier tipo, es decir, software que maneja físicamente al hardware.

**Radiofrecuencia:** es la porción del espectro electromagnético (frecuencias) que es empleado en la radiocomunicación.

**Infrarrojo:** se refiere a la radiación electromagnética con longitud de onda mayor (menor energía) a la de la luz visible por el ser humano.

**PIC:** familia de microcontroladores tipo RISC (Computador con Conjunto de Instrucciones Reducidas). [26]

**Zigbee:** es el nombre de la especificación de un conjunto de protocolos de alto nivel de comunicación inalámbrica para su utilización con radiodifusión digital de bajo consumo, basada en el estándar IEEE 802.15.4 de redes inalámbricas de área personal. [27]

**HTML5:** siendo la última versión de HTML, contiene elementos, atributos y comportamientos nuevos, ademas de un conjunto más amplio de tecnologías que proporciona mayor diversidad y alcance a los sitios Web.

**PHP (Preprocesador de hipertexto):** es un lenguaje de código abierto muy popular especialmente adecuado para el desarrollo web y que puede ser incrustado en HTML.

**SQL (lenguaje de consulta estructurado):** es un lenguaje de programación estándar e interactivo para la obtención de información desde una base de datos y para actualizarla.

**VisualBasic .NET:** es un lenguaje de programación orientado a objetos que cuenta con los beneficios que le brinda .NET Framework, el modelo de programación diseñado para simplificar la programación de aplicaciones en un entorno sumamente distribuido como lo es Internet.[28]

**Framework: Heroku: IoT: Laravel: Tarjeta de Adquisición: SQL: API: Tarea: GPIO:**

# Bibliografía

- [1] TechTarget. Internet de las cosas (iot).
- [2] Kevin Ashton. That 'internet of things' thing. 2009.
- [3] Heroku Inc. The heroku platform, 2018.
- [4] Laravel. Laravel philosophy, 2013.
- [5] MDN web docs Mozilla. Http, 2018.
- [6] MDN web docs mozilla. Http request methods.
- [7] JSON.org. Introducción a json.
- [8] Damián Pérez Valdés. ¿qué son las bases de datos?, 2007.
- [9] Aadel Howedi and Ali Jwaid. Design and implementation prototype of a smart house system at low cost and multi-functional. 2016.
- [10] Espressif Systems. Esp-wroom-32 datasheet, 2018.
- [11] Electrodragon. Esp-wroom-32.
- [12] Greenfacts. Corriente alterna y corriente continua.
- [13] CEKIT. *Curso práctico de Electrónica Industrial y Automatización*. 2001.
- [14] AMGkits. Sensor de intensidad óptica (sensor de luz) gy-30 bh1750fvi.
- [15] ElectroniLab. Sensor de temperatura y humedad dht11.
- [16] Naylamp Mechatronics. Tutorial sensores de gas mq2, mq3, mq7 y mq135.
- [17] Geekbot Electronics. Mq-135 módulo sensor de calidad del aire.
- [18] Antony Garcia Gonzales. Módulo yl-83: Un detector de lluvia, 2014.
- [19] Naylamp Mechatronics. Módulo pir hc-sr501.
- [20] Punto Flotante S.A. Pruebas al módulo sensor pasivo infrarrojo pir hc-sr501. sensor de movimiento (body motion).

- [21] Andrew S. Tanenbaum. *Sistemas Operativos Modernos*. 3ra edition, 2009.
- [22] Espressif Systems. Esp-idf programming guide.
- [23] Labcenter Electronics Ltd. Proteus design suite.
- [24] Geeky Theory. ¿qué es proteus?
- [25] Margaret Rouse. Internet de todo (ioe), 2017.
- [26] Wikipedia. Microcontrolador pic, 2017.
- [27] Zigbee. What is zigbee?, 2017.
- [28] Universidad Nacional Autonoma de Mexico. Lenguaje de programación visual basic .net, 2009.