

DOMOHELP

Convierte tu vivienda en una smart space



Memoria Premio Nacional Don Bosco



Autores: Aritz Anguera, David Zubiaurre, Egoitz Uranga, Carlos Villar y Joseba Aduriz

Tutores: David Munoz, Amaia Zabala, Xabier Izquierdo y Sonia Ortiz

Memoria DomoHelp

ÍNDICE:

1. Descripción
2. Contextualización:
 - a. Instituto de formación profesional superior Don Bosco
 - b. IES Xabier Zubiri-Manteo:
3. Justificación del proyecto
4. Objetivos
5. Descripción técnica
 - a. Apartado técnico electrónico
 - b. Apartado técnico informático

1.- Descripción

En la actualidad el ahorro de energía en el hogar es un tema que preocupa a la sociedad. A su vez, todavía hay innumerables barreras arquitectónicas (posición de los interruptores, mecanismos para subida/bajada de persianas etc.) difíciles de superar para las personas con movilidad reducida.

DomoHelp es un proyecto de colaboración entre tres alumnos del ciclo de grado superior de desarrollo de aplicaciones web de Zubiri-Manteo y dos alumnos del ciclo de grado superior de mantenimiento electrónico de Don Bosco para la creación de un sistema de domótica sencillo y barato.

Gracias a DomoHelp, por medio de una aplicación web, se podrán controlar desde un móvil, tablet u ordenador personal, diferentes dispositivos del hogar gracias a una serie de sensores y actuadores electrónicos conectados via wi-fi o diferentes protocolos a el. De este modo se facilitará la realización de operaciones como encender las luces a personas con movilidad reducida.

El concepto de dependencia se puede ampliar al ámbito social. Una persona se puede considerar socialmente dependiente cuando como consecuencia de limitaciones severas de orden físico o mental requiere la ayuda de otra persona para realizar actos vitales de la vida cotidiana. El listado de actividades más utilizado en los diferentes estudios para la clasificación y medición del grado de dependencia se basa en cuestionarios que miden la necesidad o no de ayuda de una o más personas en las actividades básicas de la vida diaria (comer, control de aparatos domésticos, andar, asearse, vestirse, bañarse) y/o instrumentales.

Existen muchas enfermedades que causan esta serie de factores de dependencia y no permiten ser independiente a una persona, lo cual es un problema que aumenta progresivamente. Nuestro proyecto está enfocado sobre todo en personas con problemas de movilidad.

Queremos aplicar la tecnología de internet de las cosas en una vivienda para así conseguir diferentes ventajas tanto de comodidad como energéticas utilizando un software de código libre llamado openhab que permite controlar aparatos de diferentes marcas en un solo sistema.

2.-Contextualización:

Instituto de formación profesional superior Don Bosco:

Don Bosco es un Instituto Específico de Formación Profesional Superior situado en Rentería (Guipúzcoa). El centro lleva ya más de 50 años formando jóvenes de cara al ámbito laboral con unos excelentes resultados. Este proyecto se llevará a cabo en el departamento de electrónica de Don Bosco, uno de los departamentos con mayor prestigio del centro, en el módulo de grado superior "Mantenimiento electrónico".

Web Don Bosco: <http://www.donbosco.hezkuntza.net>

Web Electrónica DB: <http://elektronikadonbosco.blogspot.com.es>

FacebookElectrónicaDB: <https://www.facebook.com/elektronikadonbosco>

CanalYouTubeElectrónicaDB: <http://www.youtube.com/user/elektronikadonbosco/videos>

IES Xabier Zubiri-Manteo:

IES Xabier Zubiri-Manteo es un centro público de Donostia que abarca la zona de Gros / Ulia / Ategorrieta / Egia. Oferta los niveles de enseñanza ESO, BACHILLERATO y CICLOS FORMATIVOS DE GRADO MEDIO Y SUPERIOR. Este proyecto se llevará a cabo en el departamento de informática de Zubiri, concretamente en el ciclo formativo de grado superior de Desarrollo de Aplicaciones Web

Web Zubiri Manteo: <http://www.zubirimanteo.hezkuntza.net/web/guest/inicio>

Zubiri Manteo Desarrollo Web: <http://www.zubirimanteoweb.com>

Facebook Zubiri Manteo: <https://www.facebook.com/ieszubirimanteobhi>

3.- Justificación del proyecto

Consecuencias:

La implantación y puesta en marcha de este proyecto podría tener las siguientes consecuencias a corto, medio y largo plazo:

- Controlar la vivienda desde un smartphone y facilitar ciertas tareas.
- Crear diferentes escenas que puedan hacer que ahorremos energía.
- Disminuir el gasto de luz y gas de nuestra vivienda.
- Ayudar a personas con movilidad reducida a hacer ciertas acciones más fáciles en su propia vivienda.
- Crear un servicio que antes poca gente se podía permitir por su alto coste de instalación.

4.- Objetivos:

El objetivo de este proyecto viene de la necesidad en los hogares de un ahorro de energía, sobretodo eléctrica, y de la posibilidad de ayudar a personas con algún tipo de discapacidad física a realizar ciertas tareas sin necesidad de realizar una amplia reforma.

Este sistema, gracias a una interfaz sencilla y adaptada a ellos, facilitará a personas con movilidad reducida la realización de operaciones como:

Automatizar objetos de nuestra vivienda utilizando sensores y actuadores.

- Crear diferentes entornos donde depende de la movilidad de la persona se le puede ayudar a controlar cierto aparato utilizando un móvil o un ordenador.
- Crear un servicio eficaz y barato al mismo tiempo para que cualquier familia se lo pueda permitir utilizando la tecnología de Internet de las cosas.

El usuario podrá controlar desde el exterior del recinto si se ha dejado alguna luz o electrodoméstico encendido, encender la calefacción un tiempo antes de llegar al hogar, etc...

5.- Descripción técnica:

En el apartado técnico lo dividiremos en dos partes, la parte de zubiri que sería la que se encarga del controlador remoto web y la parte electrónica de Don Bosco que se encarga de poner en marcha tanto los sensores y actuadores como el controlador de todo el sistema.

3.1.- Apartado técnico electrónico:

3.1.1.- Especificaciones

Para controlar los aparatos de la vivienda vamos a tener los nodos y para comunicarnos con el sistema central usaremos diferentes protocolos.

3.1.2.- Openhab

Para controlar y gestionar los nodos utilizaremos el programa Openhab. Openhab es un software de código libre con el que se pueden controlar aparatos de diferentes protocolos y tecnologías.

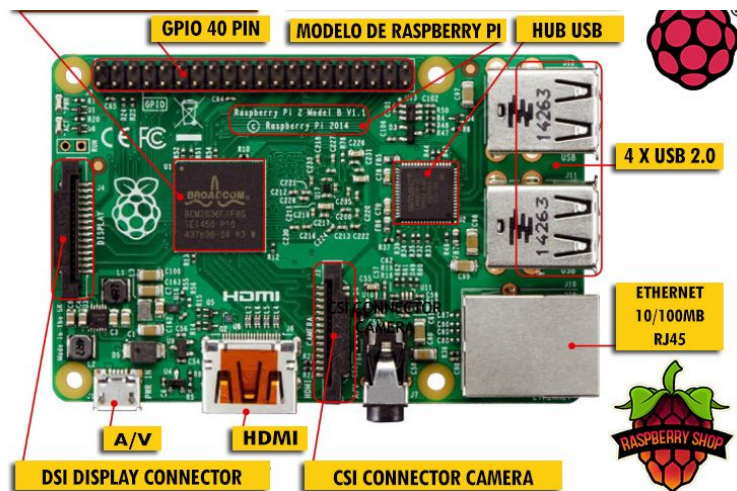
Su mayor ventaja es que el programa es de código libre, esto quiere decir que cualquier persona puede contribuir en el desarrollo del software aportando ideas y diferentes proyectos. Además puede traducir diferentes protocolos con lo que lo convierte en un producto muy fácil de añadirle diferentes productos. Para traducir estos protocolos se utiliza una aplicación llamado binding que lo que hace es traducir dicho protocolo en uno común para que lo pueda leer Openhab.



1. Imagen: OpenHab logo

3.1.3.- Raspberry

Para poner en marcha el programa openhab utilizaremos el microprocesador raspberry pi 3. Este microprocesador utiliza un sistema operativo para funcionar como si fuese un ordenador. En nuestro caso utilizaremos linux ya que es más sencillo programar con él.



2. Imagen: Partes Raspberry Pi 3

En nuestro caso utilizaremos uno de los puertos 2.0 para conectar el controlador de Z-wave. El “2.0” hace referencia a la velocidad del puerto para procesar la información. ARM1176JZF-S es el procesador que utiliza la raspberry. Tiene un procesador de 32 bits y utiliza una arquitectura de tipo risc, esto quiere decir que aprovecha su memoria al máximo para mejorar su rendimiento.

Esta placa tiene una memoria de 500Mb pero tiene la posibilidad de añadirle una tarjeta SD para guardar ahí los archivos.

La ventaja principal de ésta raspberry es su rápido procesador. La memoria RAM se guarda para acciones específicas para así tener una mayor velocidad de procesamiento en la raspberry.

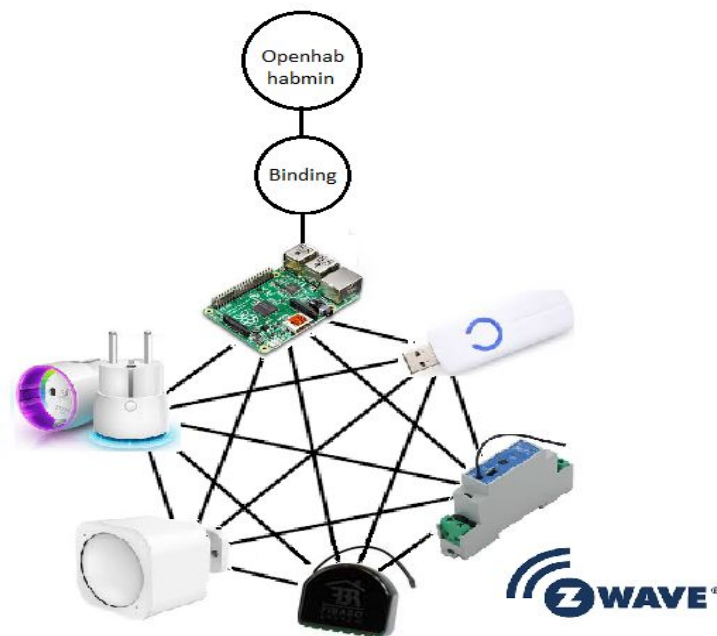
3.1.4.- Z-WAVE

Zwave es un protocolo que funciona como wifi. Su transmisión de datos se hace vía wireless es decir sin cables.

La ventaja principal de la comunicación vía wireles es el ahorro de costes que supone a la hora de hacer la instalación.

El alcance de su señal es muy grande porque cada nodo sirve de repetidor de señal. Así cada vez que se hace una ampliación del sistema no hay que tener en cuenta si la señal llegará o no.

Su comunicación es de tipo malla. Esto quiere decir que la información de cada aparato va saltando entre los diferentes nodos hasta llegar hasta el controlador de Zwave y desde ahi openhab los lee utilizando el binding anteriormente explicado.



3. Imagen: Protocolo Z-wave conectado en modo malla

3.1.5.- Z-WAVE STICK

Es el aparato que se utiliza para conectar todos los aparatos de Z-wave. Toda la información de los nodos los recibe este aparato y los transmite a openhab para que los pueda procesar.

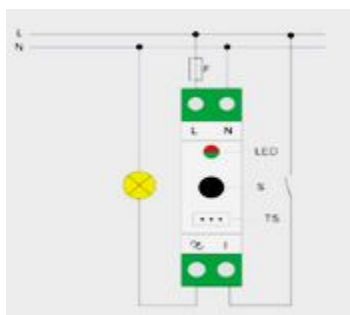
Para que openhab puede entender la información que recibe de Z-wave se utilizan los binding, en este caso el binding Z-wave y así se comunican los aparatos de Z-wave con el programa openhab.



4. imagen: Stick Z-wave

3.1.6.- QUBINO

Este aparato es un relé inteligente que mediante Z-wave puede controlar el estado de ON/OFF y también la intensidad de salida. Este aparato controlara las luminarias de la vivienda, con el se puede saber el consumo instantáneo de la luminaria y con el poder tener un control del consumo de la misma.



5. Imagen: Circuito eléctrico qubino

El Qubino se conecta a la corriente de 230V con los conectores L(linea) y N(neutro). L y Q son las salidas del Qubino, el aparato que queremos controlar lo conectaremos a

la salida Q y mediante Z-wave se podrá controlar. En la salida I pondremos un interruptor

para controlar automáticamente las luminarias. Así tendremos las dos opciones de controlar mediante el móvil o mediante un pulsador nuestras luces.

Especificaciones técnicas:

Alimentación 110-230V / 50Hz
Tensión de salida alterna: 0.85A / 230V
Tensión de salida directa: 0.85A / 30V
Márgenes de temperatura: -10°C/40°C
Konsumo: 0.7W
Alcance: 50m

3.1.7.- WALLPLUG

El wallplug es un tipo de enchufe que vía wireless se puede controlar y en consecuencia de esto se puede controlar el aparato conectado a dicho enchufe. La luz, el cargador del móvil, ... y además se puede visualizar en el programa de openhab el consumo instantáneo del aparato.

Especificaciones técnicas:

Alimentación: 110-230V / 50Hz
Corriente nominal: 11A
Potencia de salida: 2500W
Corriente máxima: 2.5A
Márgenes de temperatura: 0°C/40°C
Alcance: 50m



6. Imagen: Wallplug enchufe

3.1.8.- PHILIPS HUE

Para crear escenas utilizaremos las bombillas Philips Hue. Gracias a estas bombillas podremos controlar la iluminación de la vivienda y así elegir entre diferentes tonos de colores e intensidad de luz para diferentes momentos del día. Es decir que será diferente la iluminación si se está leyendo o si se está viendo la televisión. Además el color de la bombilla será de libre elección para el cliente. Para comunicar estas bombillas con openhab se utilizara el protocolo Zigbee que es exactamente igual que el de Z-wave solo que con diferente marca. Igual que en Z-wave tiene un sistema de malla para transmisión de datos y que cada bombilla sirve de repetidor con las demás. Las bombillas tienen un controlador llamado Bridge que se conecta a la red de la vivienda e igual que el controlador de Z-wave recibe el estado de las bombillas y así las puede controlar.

Especificaciones técnicas:

Alimentación: 230V / 50Hz

Consumo máximo : 9W

Consumo mínimo: 0.45W

Horas de vida estimadas: 15.000 ordu

Alcance: 40m



7. Imagen: Philips Hue y bridge

3.1.9.- FIBARO RELE

Es un rele que se puede controlar vía wireless utilizando el protocolo Z-wave. Además de eso se puede saber el consumo instantáneo del aparato.

Se puede utilizar para controlar luces. Si le conectamos al interruptor que controla las luces este rele podremos encender las luces automaticamente porque dependerán del estado del rele y no del interruptor.

Especificaciones técnicas:

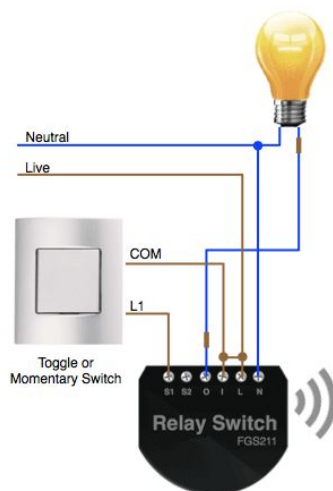
Alimentación: 110-230V / 50Hz

Consumo: 0,80W

Intensidad máxima: 6.5A

Alcance: 50m



Frecuencia: 864.8 Mhz







8. Imagen: Circuito eléctrico del fibaro rele

3.1.10.- AEOTEC 6 EN 1

Es un sensor que puede leer 6 diferentes valores y dichos valores los envía vía Z-wave a openhab para que los lea y así poder controlar diferentes valores de la vivienda como la humedad, temperatura, y consumo de la luz utilizando el sensor de presencia.

Medición	Valor	Icono
Temperatura	°C	
Humedad	%	

Presencia	ON/OFF	
Luminosidad	Lux	
Vibración	%	
Ultra Violeta	UV	

3.1.12.- APDS-9960 GESTURE SENSOR

Este sensor utiliza el chip de 8 pines APDS-9960 y los gestos los detecta mediante un led infrarrojo. Además, el sensor también puede trabajar en condiciones de poca luz, gracias al RGB digital que tiene el chip en su interior.

El sensor puede detectar 6 gestos diferentes (arriba, abajo, izquierda, derecha, lejos y cerca), el gesto detectado, a través de una señal analógica, se lo enviara a un microcontrolador, en nuestro caso el NodeMcu. Según el estado que envíe el sensor, podremos controlar diferentes dispositivos que estén situados en la casa. Por ejemplo, para apagar todas las luces podremos utilizar el gesto “Down” y para encenderlas, el gesto “Up”.

Especificaciones técnicas:

Alimentación: 3.3V

Tamaño: 18.3x16.4mm

Sensor de proximidad

Interfaz: I2C

Distancia de detección: 10-20cm

Sensor de luz de ambiente y color RGB



9. imagen: Gesture sentsorea

3.1.13.- PROTOCOLO MQTT

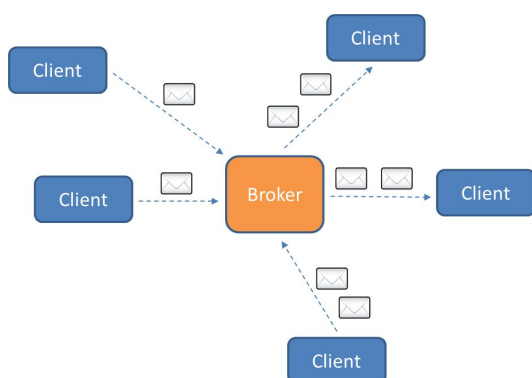
La arquitectura de una red MQTT es centralizada, es decir, tendremos un servidor central (broker) y una serie de clientes que se conectarán al broker para intercambiar mensajes. El broker es el servidor central al que se conectan los clientes y es el encargado de redirigir los mensajes al cliente adecuado. El broker es conocido como *Mosquitto*. Los clientes son los dispositivos o programas interesados en recibir y enviar datos. Un cliente puede ser una app de móvil, una aplicación de Windows, un programa en python, un arduino, openHAB...

MQTT usa el patrón Publish/Subscribe para el intercambio de mensajes. Un cliente no envía un dato directamente a otro cliente, sino que lo publica en la red MQTT y los clientes interesados en recibir ese mensaje serán notificados automáticamente por el broker. El hecho de enviar un mensaje a la red MQTT es lo que se denomina publicar (publish) un mensaje. Decirle al broker que estamos interesados en un mensaje se denomina suscribirse (subscribe) a un mensaje.

Los topics son las estructuras (directorios, direcciones postales, buzones, cuentas de twitter...) a las que publicaremos y nos suscribimos. Son un concepto abstracto, no están en ningún sitio físicamente. Tienen una estructura jerárquica, muy similar a los directorios y estructuras de carpetas de linux. Cuando hagamos un publish, lo haremos a un topic en concreto, por ejemplo: "casa/sala/temperatura". El topic será el buzón en el que dejaremos nuestro mensaje. Si el topic es el buzón, el "message" es la

información que dejamos en el mismo: la carta que lleva los datos. En un mensaje MQTT podemos enviar cualquier datos: números, strings,...

Resumiendo, protocolo con un servidor central (broker) y clientes. Los clientes envían la información a un topic y el broker reenviará el message a todos los clientes suscritos a ese topic.



10. imagen: MQTT protokoloaren eskema funtzionala

3.1.14.- ESP8266 NODEMCU

NodeMcu, es una placa de código abierto que está basada en el chip ESP8266. El ESP8266 es un microcontrolador WiFi de bajo coste producido por la empresa china Espressif. Es lenguaje de programación que utiliza es Lua, pero, al ser compatible con el Arduino IDE nosotros utilizaremos la segunda opción, porque esta, a su vez, es compatible con la librería homie-esp8266. La librería se encarga de reconectarse al router o broker en caso de caída, de suscribirse a los topics de interés o de enviar los datos estadísticos del dispositivo.

Especificaciones técnicas:

CPU: 32 bits a 80Mhz

Memoria: 64 KiB para sketch y 96 KiB de RAM

Conectividad: WiFi 802.11 b/g/n

Tipo de pines: 16 pines GPIO

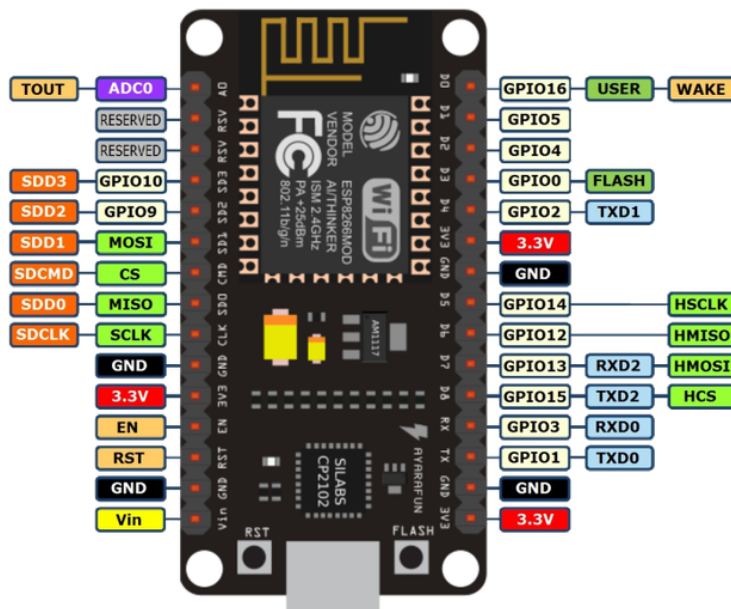
Alimentación: Micro USB 5V

Tamaño: 49 x 24.5 x 13mm



11.imagen: NodeMcu-a

Pines del NodeMcu

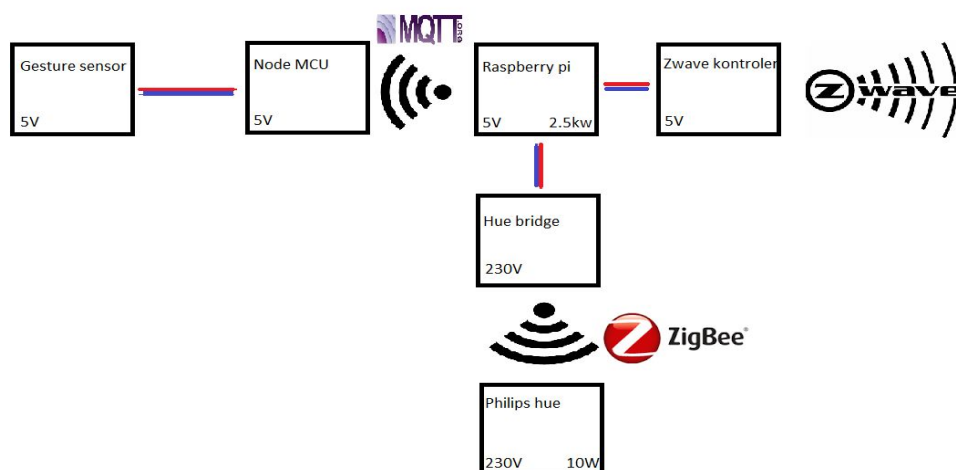


12. imagen: NodeMcu-aren pinen egitura

- Pines GPIO: estos pines pueden ser configurados como entrada o salida. Las entradas pueden ser analogicas o digitales, pero, la salida siempre sera analogica.
- Pin ADC: es un conversor que se usa para convertir la señal analogica en señal digital. Estos pines se utilizan para leer el valor de los sensores. Los sensores analogicos dan valores entre 0 y 5V, el conversor analogico digital los pasa a modo binario para que un sistema digital los pueda interpretar.
- Pines SPI: SPI es un protocolo síncrono. La sincronización y transmisión de los datos se hace mediante 4 señales; SCLK (Clock), MOSI (Master Output Slave Input), MISO (Master Input Slave Output) y CS/SELECT se utiliza para elegir al esclavo.
- Pin GND: para conectarse a “tierra”.
- Pin 3.3 V: es la tensión que estará en la salida del pin.
- Pin Vin: tensión de entrada entre 5-10V.
- Pines TX/RX: para hacer la comunicación serial.

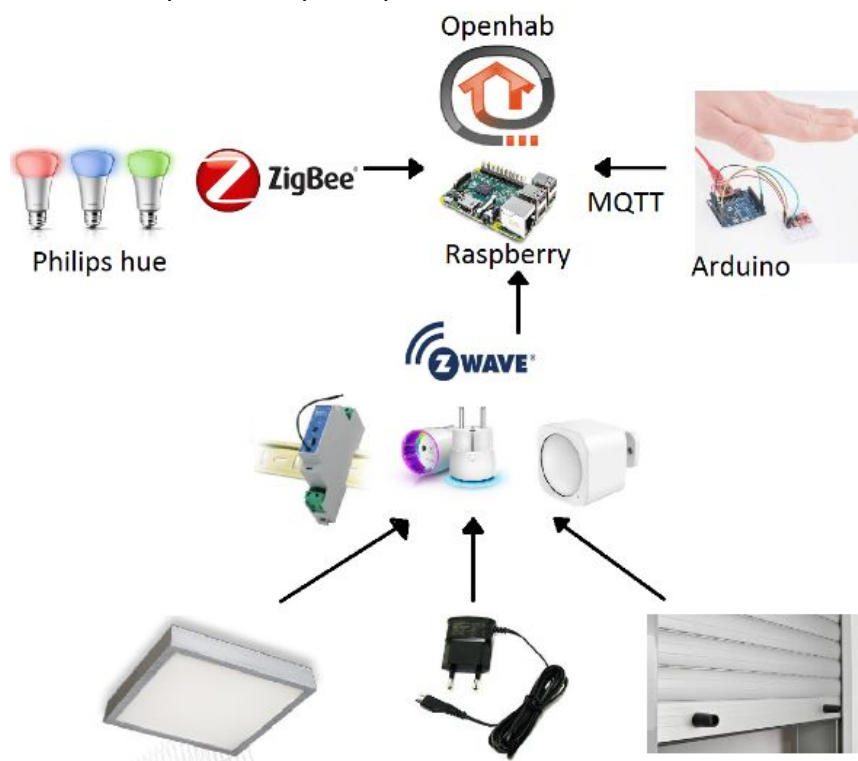
3.1.15.-Planos del proyecto

3.1.16.- Diagrama de bloques



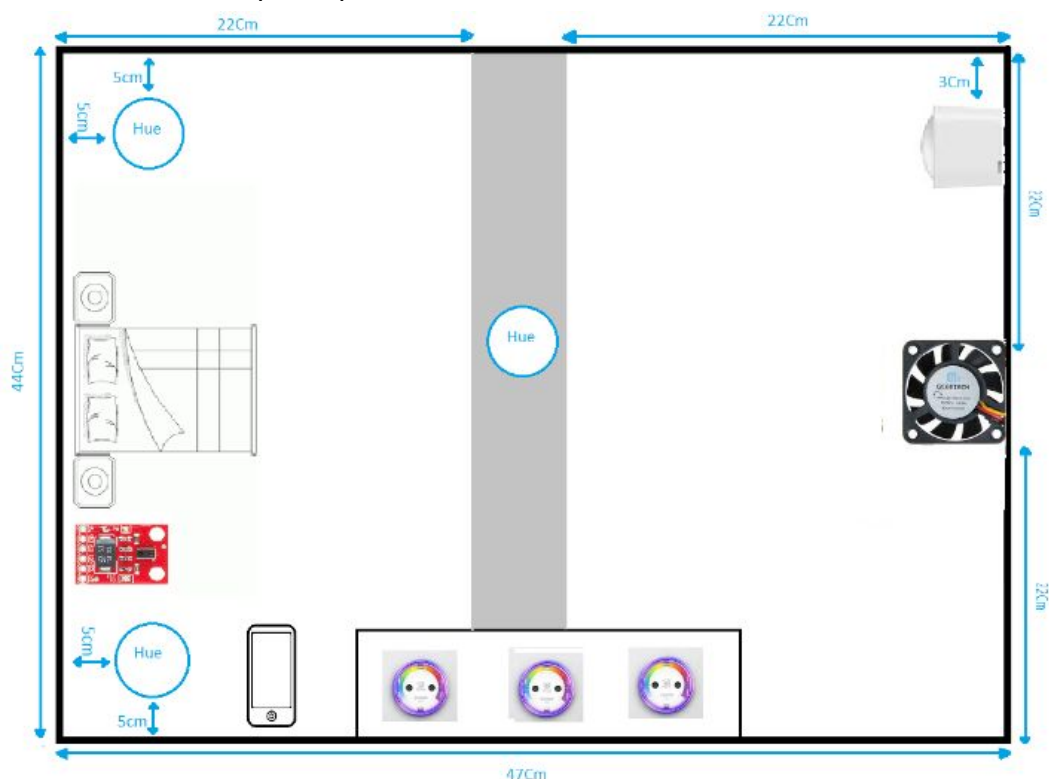
13. imagen: Diagrama de bloques

3.1.17.- Esquema del prototipo



14. imagen: Esquema del prototipo

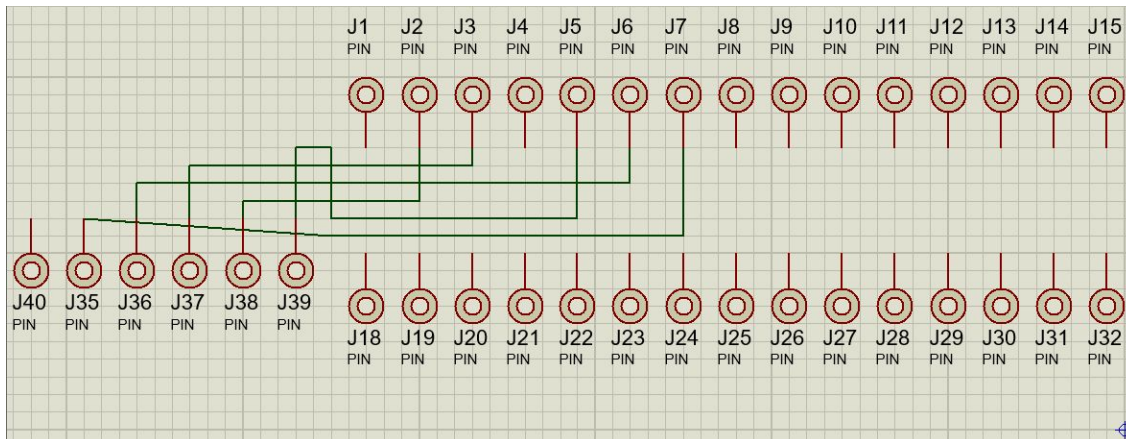
3.1.18.- Chasis del prototipo:



15. imagen: Chasis del prototipo

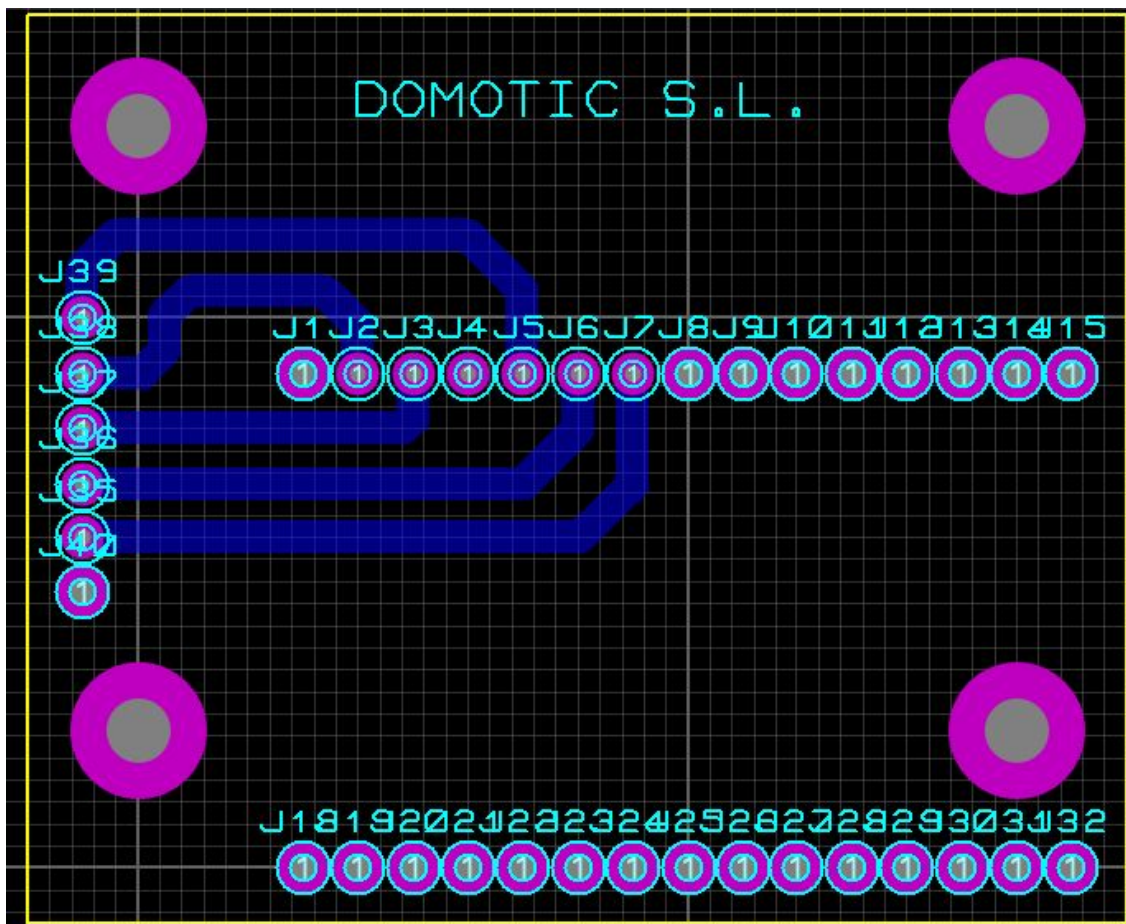
3.1.19.- Esquema eléctrico de la PCB

El circuito eléctrico que une el gesture sensor y el node MCU:



16. imagen: Esquema eléctrico de la PCB

Diseño ares del circuito eléctrico:



17. imagen: Diseño ares del circuito eléctrico

3.2.- Apartado técnico informático:

3.2.1.- Especificaciones

El desarrollo de la aplicación web se dividirá en dos partes, la parte de servidor o Back-End y la parte del cliente o Front-End.

3.2.2 Back End

La parte Back end de un sitio web es todo lo que no se ve de ella, la parte del servidor. Se encarga de interactuar con la base de datos, de las sesiones de los usuarios, y de responder a las peticiones de la parte del cliente, Front end.

En el Back end se pueden utilizar diferentes lenguajes de programación como por ejemplo: PHP, Node.js, JAVA... todos ellos son lenguajes de programación orientada a objetos. En esta área también se tiene en cuenta la seguridad del sitio web a la hora de programar, para proteger todos los datos de los usuarios etc.

Normalmente se utilizan los llamados frameworks para facilitar el trabajo y la organización a la hora de programar. Un framework es una estructura conceptual y tecnológica de soporte definido, normalmente con artefactos o módulos concretos de software, que puede servir de base para la organización y desarrollo de software.

Para el desarrollo en esta parte de la aplicación se han utilizado las siguientes tecnologías:

1. **PHP:** Es un lenguaje de programación de código del lado del servidor, que se ejecuta en el servidor. Los archivos que contienen PHP que se ejecutan en el servidor pueden acceder a bases de datos, conexiones de red y realizar otras tareas para crear la página final que verá el cliente.

18. imagen: PHP

2. **Laravel:** Es un framework de código abierto para desarrollar aplicaciones y servicios web con PHP. Tiene objetivo ser un framework que permita el uso de sintaxis elegante y expresiva para crear código de forma sencilla permitiendo multitud de funcionalidades. Aprovecha lo mejor de otros frameworks y las características de las últimas versiones de PHP. Gran parte de Laravel está formado por dependencias de otros frameworks, especialmente de Symfony.

```

151  */
152  public function edit()
153  {
154
155      $idProduct = Input::get('pid');
156
157      $produto = ProductModel::where('id', $idProduct)->first();
158
159      $nome = Input::get('eNome');
160      $descricao = Input::get('eDescricao');
161      $userID = Auth::user()->id;
162
163      $produto->nome = $nome;
164      $produto->descricao = $descricao;
165      $produto->users_id = $userID;
166
167      if (Input::hasFile('eFoto'))
168      {
169          $file = Input::file('eFoto');
170
171          $fileNameUniq = uniqid();
172          $destinationPath = public_path(). 'assets/admin/gallery/';
173
174
175          $extension = pathinfo($file->getClientOriginalName(), PATHINFO_EXTENSION);
176
177          $fileName = uniqid();
178          if(move_uploaded_file($file, 'assets/admin/gallery/'.$fileName.'.'.$extension)){
179              $produto->foto = $fileName.'.'.$extension;
180          }
181      }
182
183      $produto->save();
184
185      return Redirect::back()->with(['message' => 'Produto alterado com sucesso!']);
186  }
187
188

```

19. imagen: Laravel

3. **Pusher:** Es una herramienta que permite a la aplicación poder mostrar datos en real-time (tiempo real), sin tener que estar actualizando la página cada vez que un dato cambia de valor.

```
//hacer el pusher
$pusher = App::make('pusher');
//hacer un array con los datos a pasar
$arr = array ('estado' => $val, 'id_item' => $id);
//pasarlo a json
$datos=json_encode($arr);
//mandar mensaje
$pusher->trigger('domohelp', 'actualizar', $datos);
```

20. imagen: Pusher

4. **Base de Datos:** El desarrollo de la aplicación lo hemos hecho utilizando **MySQL**, pero ya que al desplegar surgían diferentes errores hemos tenido que migrar la base de datos a **PostgreSQL**.

PostgreSQL es un sistema de bases de datos relacional y orientada a objetos. Es una base de datos de código abierto (open source). Tiene bastante cosas en común con MySQL, y se pueden usar diferentes lenguajes.

3.2.3 Front-End

Front-End abarca todas las tecnologías que se ejecutan en la parte del cliente, es decir las que se ejecutan en el navegador si nos referimos a una aplicación web y que se resume básicamente en código desarrollado en tres lenguajes: Html, CSS y JavaScript.

1. **HTML5:** Es un estándar que sirve de referencia al software para la elaboración de la página web, define una estructura básica y un código para la definición de contenido de una página web.

El lenguaje consta de etiquetas que se escriben de este modo: **<...>** cada etiqueta tiene un significado diferente, por ejemplo, **<a>** significa que el contenido que se encuentra entre ella será un enlace, **** será una imagen etc.


```

1  <!doctype html>
2  <html>
3  <head>
4    <meta charset="utf-8">
5    <meta http-equiv="X-UA-Compatible" content="IE=edge">
6    <title>Swiffy Output</title>
7    <script type="text/javascript" src="https://www.gstatic.com/swiffy/v7.3.0/runtime.js"></script>
8    <script>
9      swiffyobject = {"as3":false,"frameRate":24,"frameCount":1,"backgroundColor":-1,"frameSize":{"ymi
10    </script>
11    <style>html, body {width: 100%; height: 100%;}</style>
12  </head>
13  <body style="margin: 0; overflow: hidden">
14    <div id="swiffycontainer" style="width: 990px; height: 90px">
15    </div>
16    <script>
17
18      var stage = new swiffy.Stage(document.getElementById('swiffycontainer'),
19        swiffyobject, {});
20      stage.start();
21    </script>
22  </body>
23 </html>

```

21. imagen: HTML

2. CSS3: Es un lenguaje de diseño gráfico para definir y crear la presentación de un documento escrito en un lenguaje marcado, como por ejemplo HTML. Normalmente se usa para establecer el diseño visual de las páginas web y de las interfaces de usuario.

Mediante CSS podemos, por ejemplo, cambiar el color de fondo de la página, colocar el texto, las imágenes... en los lugares que quieras, el tipo de letra etc.

```

/* HEADER */
#header, #container, #footer
:width 75%
:margin 0 auto
:padding 20px 40px
:margin-top 20px

#header
:background #0a3057 url(/images/logo3.png) no-repeat center center
:height 75px
:-moz-border-radius 5px
:-webkit-border-radius 5px
.right
:float right
a
:padding 3px 5px
:color #fff
&:hover
:background #000

```

22. imagen: CSS

3. JavaScript: Es un lenguaje de programación del lado del cliente ya que es el navegador el que lo procesa. Se suele utilizar para controlar las interacciones con el usuario y para crear animaciones en la página web.

```

/**
 * Receives array of numbers bigger than 0 and returns maximum value.
 * @param {array} incoming array of numbers
 * @return {number} The maximum value in array
 */
function getMaxValue(array){
    // Declare a new variable to hold maximum value
    var maxValue = 0;

    // Iterate over array's elements
    for (var index=0; index<array.length; index++){
        // Get array element at specific index
        var element = array[index];

        // In case element value is bigger than current maxValue - update current maxValue
        if (element > maxValue) {
            maxValue = element;
        }
    }

    // Return maxValue
    return maxValue;
}

var myArray = [1,25,3,94,5,12,4,5,1,123,524,123];
var maxValue = getMaxValue(myArray);
console.log("maxValue == " + maxValue);

```

23. imagen: JavaScript

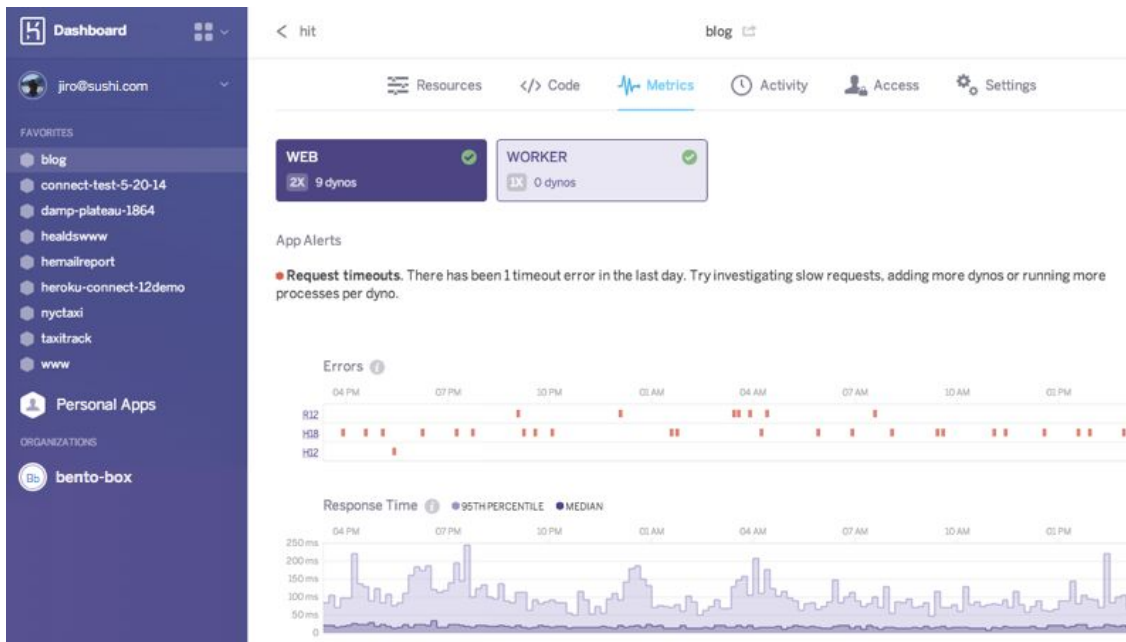
4. MaterializeCSS: es un framework de diseño basado en Material Design creado por Google. Este contiene diferentes componentes para facilitar el diseño de la web. También hace que la página se adapte al dispositivo en el cual se está visualizando, a ese tipo de diseño se le conoce como **responsive design**.

3.2.4. Despliegue

Como plataforma de despliegue hemos elegido Heroku <https://www.heroku.com/>

Heroku es una plataforma como servicio de computación en la Nube que soporta distintos lenguajes de programación.

Heroku es propiedad de Salesforce.com. Heroku, es una de las primeras plataformas de computación en la nube, que fue desarrollada desde junio de 2007, con el objetivo de soportar solamente el lenguaje de programación Ruby, pero posteriormente se ha extendido el soporte a Java, Node.js, Scala, Clojure y Python y (no documentado) PHP. La base del sistema operativo es Debian o, en la nueva plataforma, el sistema basado en Debian Ubuntu.



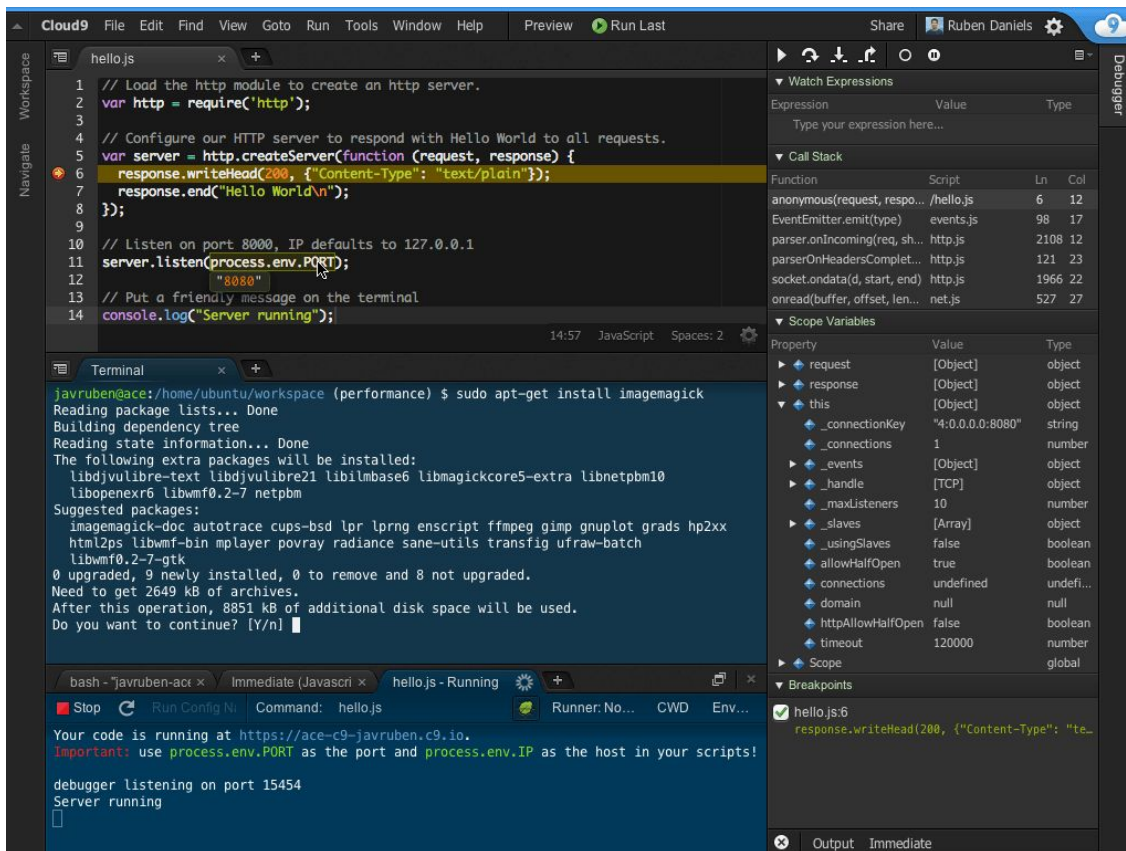
24. imagen: Plataforma de despliegue

3.2.5. Herramienta de desarrollo

Para trabajar en el desarrollo hemos elegido Cloud9 <https://c9.io/>.

C9.io es un entorno de desarrollo integrado online de código abierto. Soporta cientos de lenguajes de programación. Permite a los desarrolladores empezar a programar inmediatamente con entornos de trabajo pre configurados, los cuales puedes compartir con otros compañeros.

También contiene la característica de pre visualizar y testear la compatibilidad del código escrito con diferentes navegadores.



25. imagen: Cloud9

3.2.6 Implementaciones futuras

Para el futuro, con el fin de acercarnos más a las personas con movilidad reducida nos gustaría implementar un control de voz para la interfaz de usuario así como también hacer que el usuario pueda usar el interfono de casa si usa un dispositivo móvil o ver la imagen de la cámara del interfono del portal.