Universidad Católica de Santa María

Facultad de Ciencias e Ingenierías Físicas y Formales

Escuela Profesional de Ingeniería de Sistemas

"PROTOTIPO DE CONTROL DOMÓTICO USANDO ARDUINO MEGA, PATRON MVC Y SERVIDOR IIS ORIENTADO A DAR AYUDA A PERSONAS EN SILLA DE RUEDAS"

Tesis presentada por el Bachiller:

López Villegas, Juan Alonso

para optar por el Título Profesional de:

Ingeniero de Sistemas

Asesor:

Ing. José Sulla Torres

AREQUIPA-PERÚ

2018

FACULTAD DE CIENCIAS E INGENIERIAS FISICAS Y FORMALES

ESCUELA PROFESIONAL DE INGENIERIA DE SISTEMAS

INFORME DICTAMEN DE BORRADOR DE TESIS

VISTO

El Borrador de Tesis titulado:

Prototipo de sistema domótico usando Arduino Mega, Patrón MVC y Servidor IIS orientado a mejorar la calidad de vida de personas en silla de ruedas.

Presentado por (el) (la) (los) Bachiller (es):

Juan Alonso Lopez Villegas

Nuestro dictamen es:

Procedente

OBSERVACIONES: Ninguna

Arequipa, 19 de setiembre de 2018

José Sulla E.
1635

1221

AREQUIPA - PERÚ

FACULTAD DE CIENCIAS E INGENIERIAS FISICAS Y FORMALES

PROGRAMA PROFESIONAL DE INGENIERIA DE SISTEMAS

FORMATO DE OBSERVACIONES DE EJEMPLAR FINAL

El ejemplar final de Tesis/Trabajo Informe, titulado:

PROTOTIPO DE SISTEMA DOMOTICO USANDO ARDUINO MEGA, PATRON MVC Y SERVIDOR HS ORIENTADO A MEJORAR LA CALIDAD DE VIDA DE PERSONAS EN SILLA DE RUEDAS

Presentado por el (los) Titulando (s):

LOPEZ VILLEGAS JUAN ALONSO

Tiene las siguientes observaciones:

1) REHACER ~~CONCLUSIONES~~ RECOMENDACIONES
2) ARREGLAR 3RA CONCLUSIÓN
3) ORIENTAR A CONTROL DOMÓTICO EL ALCANCE
4) PONER CARACTERÍSTICAS DE LOS ENCUESTADOS
5) MEJORAR LA PARTE DE VALIDACIÓN
6) REVISAR ORTOGRAFÍA
7) REVISAR NUMERACIÓN Y MAPEO DE FIGURAS Y TABLAS
8) AUMENTAR MEJORAR LA CALIDAD DE VIDA

Arequipa, 17 OCTUBRE 2018

Presidente	Integrante	Secretario

Levantadas las observaciones formuladas anteriormente, se autoriza la impresión y empastado del ejemplar final

Arequipa,.....

Presidente	Integrante	Secretario

"IN SCIENTIA ET FIDE EST FORTITUDO NOSTRA"
En la Ciencia y en la Fe está nuestra Fuerza

Arequipa 02 de noviembre del 2018

OFICIO Nº 103-EPIS-2018

Señora Mgter
AYMEE PEREZGOMEZ
Coordinadora de Centro de
Informacion y Bibliotecas
Presente.-

Previo cordial saludo me dirijo a usted a fin de hacer de su conocimiento que en el momento de la sustentación se hizo con el siguiente título Tesis:

"PROTOTIPO DE SISTEMA DOMOTICO USANDO ARDUINO MEGA, PATRON MVC Y SERVIDOR HS ORIENTADO A MEJORAR LA CALIDAD DE VIDA DE PERSONAS EN SILLA DE RUEDAS"

Creo necesario indicar que a pedido de todo el jurado en su totalidad se sugirió el cambio solo en el Titulo de la tesis, siendo el contenido el mismo, el cual quedara queda inscrito con el siguiente nombre:
"PROTOTIPO DE CONTROL DOMOTICO USANDO ARDUINO MEGA, PATRON MVC Y SERVIDOR IIS ORIENTADO A DAR AYUDA A PERSONAS EN SILLA DE RUEDAS"

presentado por el (los) Bachiller:

- **LOPEZ VILLEGAS JUAN ALONSO**

Para optar el Título Profesional de	:	**INGENIERO DE SISTEMAS:**
Fecha de Sustentación	:	17 DE OCTUBRE DEL 2018
Aprobado por	:	**UNANIMIDAD**
Folio	:	**068**
Tomo	:	**V**

Sin otro particular, le reitero los sentimientos de mi consideración y estima personal.
Atentamente,

UNIVERSIDAD CATÓLICA DE SANTA MARÍA
Dr. GUILLERMO CALDERÓN RUIZ
Director de la Escuela Profesional de
Ingeniería de Sistemas

PRESENTACIÓN

Señor Director de la Escuela Profesional de Ingeniería de Sistemas de la Universidad Católica de Santa María.

Señores Miembros del Jurado Dictaminador.

De conformidad con las disposiciones del reglamento de Grados y Títulos de la Escuela Profesional de Ingeniería de Sistemas, pongo a vuestra consideración el presente trabajo de investigación titulado: "**PROTOTIPO DE CONTROL DOMÓTICO USANDO ARDUINO MEGA, PATRON MVC Y SERVIDOR IIS ORIENTADO A DAR AYUDA A PERSONAS EN SILLA DE RUEDAS"**, el mismo que de ser aprobado me permitirá obtener el Título Profesional de Ingeniero de Sistemas.

Juan Alonso López Villegas

INDICE.

ÍNDICE DE FIGURAS

INDICE DE TABLAS

RESUMEN

Se ha diseñado un sistema Web con una interfaz sencilla la cual puede ser ejecutada desde cualquier dispositivo que tenga un navegador Web y conexión Wifi.

Con este sistema se puede controlar el abrir y cerrar de puertas, ventanas, persianas, encender y apagar luces, activar o desactivar el sistema de climatización - ventilación, activar alarmas de seguridad, control de ascensores que lo desplacen de piso en piso, contar con sensores que capturen señales del medio ambiente y estos sirvan como una alerta a la persona con discapacidad y esta pueda tomar una decisión por sí misma y poder ejecutarla, etc. Es decir tener todo el control del hogar en sus manos.

Se ha usado en el presente proyecto sensores, circuitos electrónicos y hardware de bajo costo como lo es Arduino, los cuales se pueden modificar para que funcionen en la vida real sin ningún problema.

El resultado final es la obtención de un sistema de control que permite el control de dispositivos mecánicos y electrónicos de un hogar de manera segura y sencilla.

En conclusión, el presente proyecto dará una visión distinta al enfoque tradicional de la domótica; hay muchos estudios sobre casas inteligentes, pero observamos que no hay suficientes proyectos domóticos para ayudar a las personas con discapacidad.

Palabras Clave: Sistema Web, discapacitados, Arduino.

ABSTRACT

A Web system with a simple interface has been designed which can be executed from any device that has a Web browser and Wifi connection.

With this system you can control the opening and closing of doors, windows, blinds, turn lights on and off, activate or deactivate the HVAC system, activate security alarms, control elevators that move from floor to floor, have sensors that capture signals from the environment and these serve as an alert to the person with disabilities and they can make a decision on their own and be able to execute it, etc. In other words, have all the control of the home in your hands.

It has been used in this project sensors, electronic circuits and low cost hardware such as Arduino, which can be modified to work in real life without any problem.
The final result is the obtaining of a control system that allows the control of mechanical and electronic devices of a home safely and easily.

In conclusion, the present project will give a different vision to the traditional approach of home automation; There are many studies on smart homes, but we note that there are not enough home automation projects to help people with disabilities.

Keywords: Web system, disabled, Arduino.

INTRODUCCIÓN

Hoy en día la tecnología está en todas partes, desde cuando queremos ver nuestros programas en un televisor SMART de última generación hasta cuándo vamos a realizar una transacción bancaria vía Internet.

El propósito de este proyecto es aplicar la tecnología para que de alguna manera se pueda automatizar las tareas básicas a las que se enfrenta a diario una persona en silla de ruedas, para tratar de mejorar la usabilidad de dispositivos de uso cotidiano.

Este proyecto se da con el fin de crear cierta independencia para estas personas que no pueden movilizar la mayoría de su cuerpo o sufren de alguna discapacidad, las cuales, dependiendo de otras, logran desarrollar con dificultad sus diferentes actividades diarias.

Es posible construir una aplicación Web utilizando software y hardware que está al alcance de cualquier persona debido a su bajo costo, logrando el funcionamiento de esta aplicación en cualquier dispositivo móvil o televisor SMART que podrá permitir la automatización y control de prácticamente todo el hogar.

El prototipo cubre las 3 necesidades de un sistema domótico (Recuero, 2008)

- Necesidades de seguridad, prevención de accidentes corporales y materiales: el prototipo incluye un sensor de gas que detecta fugas, y un sensor de intrusión; ambos activan alarmas.
- Necesidades de comodidad ambiental, que implican la creación de un medio ambiente agradable: el prototipo incluye control de calefacción y ventilación.

- Necesidades de comodidad de actividad, que provienen del deseo de facilitar las actividades cotidianas: El prototipo facilita la apertura/cierre de puertas, ventanas, control de ascensor, luces, etc.

El presente trabajo consta de los siguientes capítulos:

El capítulo 1 trata sobre las generalidades del trabajo de investigación, se listan los objetivos de la investigación, indicando su objetivo general y sus objetivos específicos mientras que el capítulo 2 enmarca el aspecto conceptual de la tesis. En el capítulo 3 se detalla los aspectos económicos y comerciales del proyecto así mismo se define y describe la metodología del proyecto así como también la descripción de la aplicación tanto en software como en hardware.

En el capítulo 4 se muestran los resultados obtenidos y finalmente se arriba a las conclusiones y mejoras al proyecto.

CAPÍTULO 1: PLANTEAMIENTO TEÓRICO

1.1. Titulo Descriptivo

"Prototipo de control domótico usando Arduino Mega, patrón MVC y servidor IIS orientado a dar ayuda a personas en silla de ruedas"

1.2. Definición del Problema.

Sufrir de algún tipo de discapacidad definitivamente pone en desventaja, debido a que la persona que sufre este mal debe de depender en menor o mayor proporción de otra, esto trae consigo la depresión, pérdida de autoestima, etc. que posteriormente traerá como consecuencia trastornos psicológicos más graves.

El solo hecho de pensar que es posible romper este esquema en la actualidad usando la tecnología, nos hace pensar en las infinitas posibilidades para lograr ayudar a personas que se encuentran postradas en una silla de ruedas.

Con el presente proyecto se pretende mejorar las limitaciones que tiene una persona en silla de ruedas logrando la automatización de la mayoría de dispositivos eléctricos/electrónicos y mecánicos del hogar facilitando y mejorando de esta manera la usabilidad de éstos dispositivos de uso cotidiano, esto gracias al uso de circuitos simples, hardware muy económico y software libre.

En la figura 1 se muestra un resumen de manera global de la cantidad de personas con discapacidad en nuestro país según el Instituto Nacional de Estadística e Informática:

Figura 1: *Resultados de primera encuesta nacional sobre discapacidad.*

Fuente: *INEI (INEI, 2017)*

En Perú el número de personas discapacitadas corresponde al 5,2% de la población del país (INEI, 2017), es decir que casi 2 millones de personas de un total de 22 millones de Peruanos, presentan algún tipo de discapacidad permanente.

Según (SCHOTT, 2005)**,** si bien, cada discapacidad plantea una serie de retos distintos en el quehacer cotidiano a la persona que la padece, sin duda alguna que una de las problemáticas más recurrentes, sobre todo en los casos de personas afectadas con discapacidades asociadas al desplazamiento, es la de la interacción del individuo con su entorno, esto se debe principalmente a la poca o difícil

movilidad asociada con el reposo, fracturas o vejez, o al uso de aparatos como sillas de ruedas, muletas, andadores, etc.

1.3. Alcances

- Controlar diferentes dispositivos eléctricos, electrónicos y mecánicos en el hogar.
- Conocer y aplicar el patrón MVC.
- Implementar la validación de permisos y accesos al sistema.
- Lograr la seguridad en el hogar frente a actos delictivos

1.4. Línea y Sub-línea de Investigación

- Sistemas de información
- Tecnología de información y comunicaciones

1.5. Objetivos

1.5.1. Objetivo General

Desarrollar un prototipo de control domótico usando Arduino Mega, patrón MVC y servidor IIS orientado a dar ayuda a personas en silla de ruedas

1.5.2. Objetivos Específicos

- Construir e implementar una estructura electrónica.
- Desarrollar e implementar un sistema web adaptable (sistema responsivo).
- Controlar y automatizar los diferentes dispositivos de uso cotidiano.

1.6. Descripción del Proyecto a medio y largo plazo

El presente proyecto necesita de la intervención de varias ramas de la Ingeniería, tales como Mecánica, Electrónica, Sistemas, Arquitectura, Eléctrica, etc.; a mediano plazo se puede generar alianzas entre las diferentes facultades para poder desarrollar un proyecto completo que sea factible y fiable, esto ayudaría a reducir los costos de producción.

A largo plazo, al ser este un proyecto social, la intervención de alguna ONG puede dar mayor impulso con su apoyo financiero, logrando que el proyecto se lleve a producción implementándose en una casa; una vez que se vean las ventajas de este sistema, podría expandirse a nivel regional y nacional.

1.7. Usuarios del Proyecto.

Los usuarios del sistema propuesto serán las personas que sufran de alguna discapacidad que no le permita de alguna manera la movilización de sus piernas o brazos y que se movilicen a través de una silla de ruedas.

Para el funcionamiento y manipulación del sistema se necesita por lo menos tener visión al menos en un ojo y poder mover mínimo una mano.

1.8. Beneficios.

Los beneficios van a ser inmediatos debido a que en tiempo real el usuario del sistema podrá controlar de manera sencilla los diferentes componentes que automatizan el hogar.

Los beneficiados directos serán las personas con discapacidad que usen silla de ruedas, posiblemente atraiga a instituciones sin fines de lucro interesadas en la

ayuda social. Con el desarrollo del proyecto se tendrá una imagen positiva, humana y sensible, que se interesa por la población que se encuentra en desventaja.

1.9. Riesgos que debemos afrontar.

No tecnológicos: la implementación del cableado e instalación de equipos grandes como ascensores, pueden traer consigo problemas en su ubicación o adaptación a la actual construcción del hogar.

Se debe de tener especial cuidado en la seguridad de nuestra red interna, así mismo en la confidencialidad respecto a las contraseñas usadas para el ingreso a este sistema.

CAPÍTULO 2: MARCO TEÓRICO.

2.1. Estado del Arte

- En RIVERA (2016) se describe el desarrollo tecnológico de un sistema de control domótico con la finalidad de brindar ayuda a personas discapacitadas que padecen paraplejia y cuadriplejia mediante la captura e interpretación de señales electroencefalográficas.

  Las señales detectadas por el sensor de electroencefalografías son enviadas por medio de comunicación Bluetooth al sistema de control en la PC donde son analizadas para poder detectar las variaciones electroencefalográficas que genera el cerebro humano; al cambiar de estado de ánimo y generar parpadeo, muestra en pantalla al dispositivo realizando su función, al detectar que el usuario final ha parpadeado y enfoca su atención en el dispositivo a controlar, éste cambia de estado.

- Según Camargo (2012), realiza la descripción de un sistema de reconocimiento autónomo del habla (RAH), el cual permite registrar, validar, identificar y generar acciones a partir de palabras o frases capturadas por un micrófono, esta señal de voz se convierte a un lenguaje digital y se divide en muestras para su posterior procesamiento utilizando diferentes algoritmos.

  Posteriormente haciendo uso del protocolo Zigbee y un emisor Xbee, se envía esta señal digital a un receptor que consta de una circuitería y de un receptor Xbee ejecutando una acción, pudiendo ser el control de una luz.

- Según se expone en Bonino (2012), el dispositivo de control domótico usado es un reloj pulsera llamado “dWatch”, éste es un dispositivo personal y de control portátil integrado en una plataforma inteligente para sistemas domóticos que permite a los usuarios realizar diversas operaciones en sus hogares y recibir notificaciones del medio ambiente, esto beneficia en particular a personas que no están acostumbradas a sistemas informáticos (por ejemplo, ancianos) o en contextos donde los usuarios no están frente a una pantalla.

  Las ventajas de usar un reloj como dispositivo de control es que a diferencia de las tablets o teléfonos, este no se tiene que encender antes de acceder y usar el dispositivo.

  Otra ventaja es que se puede usar en la ducha o con las manos mojadas, además de que una gran parte de la población ya está acostumbrada a usar relojes.

  Internamente el dWatch incluye una placa principal con un microprocesador, posee sensores, tarjeta inalámbrica y un firmware.

  El dWatch permite al usuario controlar electrodomésticos, regular fácilmente la temperatura interna del hogar, encender una alarma en caso de emergencia o como recordatorio, recibir notificaciones sobre eventos y demás.

  El dWatch, por defecto solo muestra la hora y la fecha pero, al presionar uno de los botones del reloj, se puede ingresar al menú principal y realizar diferentes funciones. Por ejemplo, cuando se siente calor, se selecciona el menú "temperatura" y puede bajar la temperatura en la habitación al presionar el botón de flecha hacia abajo en el reloj. También se dispone de sensores de alarma para detectar incendios mostrando para este caso en la pantalla del reloj “fuego.cocina”.

- Según Castellina (2011) expone el diseño y desarrollo de DOGeye, uno de los primeros hogares controlados por el seguimiento de ojos diseñado especialmente para pacientes que sufren de la enfermedad degenerativa conocida como ELA (esclerosis lateral amiotrófica)

  Este proyecto permite la gestión y control domótico del hogar utilizando el seguimiento ocular.

  El modo de funcionamiento es el siguiente, generalmente el ojo no se mueve suavemente sobre el campo visual; por el contrario, se mueve haciendo unos saltos rápidos llamados movimientos sacádicos, estos movimientos duran de 30 a 120 ms y son analizados usando la técnica de reflexión corneal que consiste en enviar un pequeño haz infrarrojo hacia el centro de la pupila , esta luz se refleja en los ojos las cuales son captadas por las cámaras del controlador ocular, luego mediante el filtrado y el cálculo, el controlador identifica donde está mirando.

  Para realizar el click del mouse se utiliza la técnica de permanencia la cual consiste en centrar los ojos en un área específica durante un predeterminado número de milisegundos lo que ocasiona un click.

  Las personas con discapacidades graves pueden obtener grandes ventajas de los sistemas de seguimiento ocular para controlar sus hogares, ya que generalmente conservan el control normal de sus ojos.

  Este sistema permite controlar los dispositivos básicos presentes en una casa como persianas, puertas, lámparas, etc., también permite manejar el sistema de calefacción y enfriamiento de la casa así como también control de cámaras y alarmas contra robos.

- Según Ziyu Lv (2012), se describe el diseño del proyecto iCare para aquellos adultos mayores que viven independientemente en sus propios hogares, ya que

existe un riesgo creciente de caídas y accidentes que podrían amenazar sus vidas.

El proyecto consiste en el desarrollo de un sistema de monitoreo de salud móvil llamado iCare para personas mayores el cual puede ser usado en cualquier momento y lugar.

Se usa sensores corporales inalámbricos y teléfonos inteligentes para monitorear el bienestar de los ancianos. Puede ofrecer monitoreo remoto para personas mayores en cualquier momento en cualquier lugar y proporcionar servicios personalizados para cada persona en función de su estado de salud personal. Al detectar una emergencia, el teléfono inteligente alertará automáticamente a las personas previamente asignadas que podrían ser familiares y amigos de los ancianos, y llamará a la ambulancia del centro de salud. También el sistema actúa como orientador médico ya que ofrece una base de datos de conocimientos médicos para que la familia o personas cercanas al anciano pueda servir de ayuda al médico.

El sistema también presenta algunas funciones como recordatorios periódicos, alarmas rápidas, etc.

- Según CasaDomo (2018) se muestra la domótica para discapacitados en el mundo, en la cual la compañía “B&J Adaptaciones” que fue fundada por Joaquín Romero quien sufre de esclerosis múltiple, con la ayuda de su hermano Borja Romero, Ingeniero Técnico de Telecomunicaciones quienes con la finalidad de mejorar la calidad de vida y la autonomía de las personas con discapacidad mediante la tecnología, crearon un sistema domótico especial para discapacitados, en el cual lo más resaltante es que utilizan mecanismos como

una grúa la cual permite levantar a la persona de la cama y sentarla en la silla de ruedas, así como también de la Telesilla que permite ducharse o ir al inodoro de forma autónoma, sin precisar ningún tipo de ayuda. La transferencia de la grúa a la telesilla se realiza con total seguridad, sin riesgo de caídas.

2.2. Bases Teóricas del proyecto

2.2.1. Domótica

Se denomina domótica a la serie de sistemas integrados e interrelacionados que se instalan en un hogar y que permiten la automatización del mismo y su control tanto desde dentro de la casa como desde fuera. (Chaparro, 2013)

Entre otros servicios, la domótica aporta gestión de energía, comunicaciones y seguridad, con la misión de brindarles a los habitantes del hogar bienestar y comodidad. Cabe indicar que popularmente se denomina a estas casas como hogares inteligentes. (Eklund, 2013)

En la figura 2 se representa a una casa inteligente, detallando las diferentes acciones que se pueden realizar en ella.

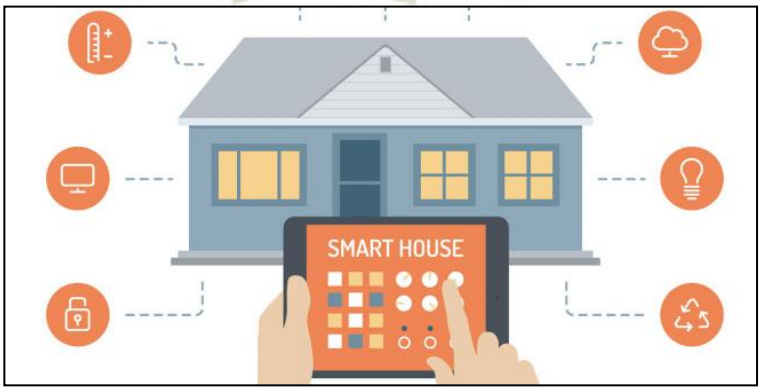

Figura 2: *Representación de un sistema de automatización en un hogar.*

Fuente: *definición ABC* (La domótica en nuestra vida vía Definición ABC)

2.2.2. Arduino.

Arduino es una plataforma electrónica de código abierto basada en hardware y software fáciles de usar. (DAVID, 2013)

Arduino consta de una placa principal, en la cual se encuentran conectados todos los componentes electrónicos que gestionan los demás circuitos ensamblados.

El modelo de Arduino usado en el proyecto es el Arduino Mega, que como sus homólogos, está basado en el micro controlador Atmega 2560 (figura 3).

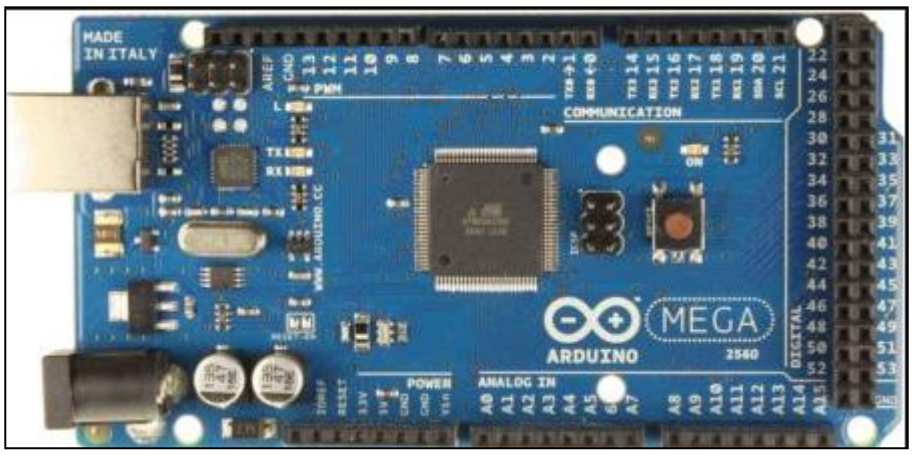

Figura 3: *Placa Arduino Mega basada en chip Atmega 2560.*

Fuente*: Panamahitek (Panamahitek, 2017)*

La IDE de Arduino es un entorno de desarrollo el cual permite la programación de la placa, tiene como base el entorno de Processing y Wiring (open source).

Tiene instalado como base el cargador de arranque (bootloader), el cual se ejecuta en el microcontrolador.

La plataforma Arduino se programa mediante el uso de un lenguaje propio basado en el lenguaje de programación de alto nivel Processing que es similar a C++. (Arduino, 2018)

Más adelante se estará detallando la distribución y especificación de pines usados en el presente proyecto.

a) Especificaciones técnicas del Arduino Mega. (GONZALES, 2013)

Arduino Mega posee las siguientes especificaciones

- Microcontrolador: ATmega2560
- Voltaje Operativo: 5V
- Voltaje de Entrada: 7-12V
- Voltaje de Entrada(límites): 6-20V
- Pines digitales de Entrada/Salida: 54 (de los cuales 15 proveen salida PWM)
- Pines análogos de entrada: 16
- Corriente DC por cada Pin Entrada/Salida: 40 mA
- Corriente DC entregada en el Pin 3.3V: 50 mA
- Memoria Flash: 256 KB (8KB usados por el bootloader)
- SRAM: 8KB
- EEPROM: 4KB
- Clock Speed: 16 MHz

El Arduino Mega se basa en el Atmega2560. Cuenta con 54 pines digitales de entrada / salida (de los cuales 15 se pueden utilizar como salidas PWM), 16 entradas analógicas, 4 UARTs (puertos serie de hardware), un oscilador de 16MHz, una conexión USB, un conector de alimentación, un conector ICSP, y un botón de reset.

Puede ser alimentado por USB o con una fuente de alimentación externa, se recomienda entre 7V a 12V, si sea alimenta con más de 12V, se corre el riesgo de sobrecalentar la placa.

b) Memoria del Arduino Mega

El Atmega2560 tiene 256 KB de memoria flash para almacenar el código, de esta cantidad se utiliza 8Kb para el arranque del Arduino o Bootloader, tiene 8Kb de SRAM y 4Kb de memoria EEPROM.

Cada uno de los 54 pines digitales del Arduino Mega se puede utilizar como una entrada o como una salida, utilizando las funciones pinMode(), digitalWrite() y digitalRead(). Operan a 5 voltios. Cada pin puede proporcionar o recibir 20 mA. En la figura 4 se nuestra la distribución y descripción de cada grupo de pines del Arduino Mega.

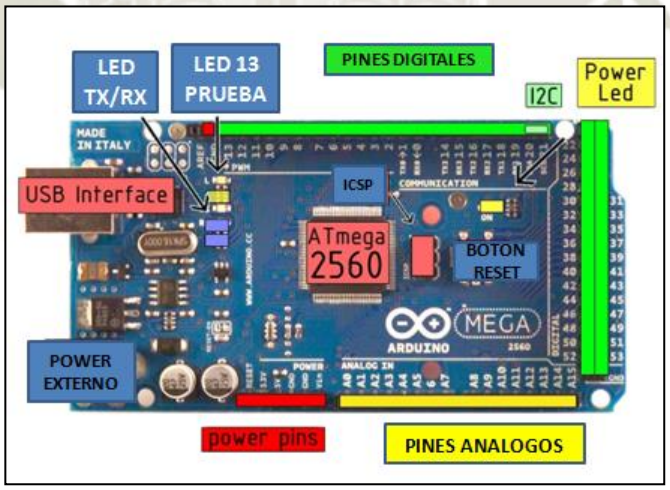

Figura 4: *Distribución de pines de Placa Arduino Mega.*

Fuente*: Arduino (Arduino, 2017)*

2.2.3. Wi-Fi.

Wi-Fi es una tecnología de comunicación que permite la conexión e intercambio de información entre tablets, computadoras, laptops, celulares, televisores, etc. según se representa en la figura 5.

Figura 5: *Wi-fi en cada rincón de nuestro hogar.*

Fuente*: Cadenatres (cadenatres, 2017)*

Esta conexión se realiza a través de uno o varios puntos de acceso inalámbricos, usando ondas de radio las cuales son captadas por los clientes para el posterior intercambio de información. Se basa en el estándar IEEE 802.11 y permite la conexión con cualquier dispositivo que posea un controlador de interfaz de red inalámbrica.

Para el proyecto se usará Wi-fi para conectar los diferentes dispositivos con los cuales se desee ingresar al sistema de login que permitirá el ingreso al sistema principal; estos dispositivos pueden ser: celulares inteligentes, tablets, computadoras, laptops o televisores del tipo Smart.

2.2.4. Servidor Web IIS.

Según Gonzales, (2006) El Servidor Web IIS engloba un conjunto de herramientas destinadas al control de servicios de Internet como el Web, FTP, correo y servidores de noticias. Además incluye el soporte necesario para la creación de páginas dinámicas en el servidor mediante el lenguaje ASP.

Los servicios de Internet Information Server (IIS) simplifican la publicación de información en Internet o en la Intranet. IIS incluye una amplia gama de funciones administrativas para controlar sitios Web y el servidor Web.

El proyecto consta de una aplicación Web responsiva, el servidor Web IIS alojará a la aplicación principal, la cual ejecutará los comandos enviados al Arduino, éste los procesará y realizará las diferentes acciones en los diferentes ambientes controlando los diferentes dispositivos como puertas, ventanas, sensores, etc.

2.2.5. Visual Studio Community 2013.

En Mcrosoft (2013) se indica que Visual Studio es un conjunto completo de herramientas de desarrollo para la generación de aplicaciones Web ASP.NET, Servicios Web XML, aplicaciones de escritorio y aplicaciones móviles. Visual Basic, Visual C# y Visual C++ utilizan todos el mismo entorno de desarrollo integrado (IDE), que habilita el uso compartido de herramientas y facilita la creación de soluciones en varios lenguajes. Asimismo, dichos lenguajes utilizan las funciones de .NET Framework, las cuales ofrecen acceso a tecnologías clave para simplificar el desarrollo de aplicaciones Web ASP y Servicios Web XML

El proyecto Web a presentar se encuentra desarrollado en C# con características adaptables (responsivo).

2.2.6. Sensores

Según se explica en ARDUINO (2013), Un sensor es un aparato capaz de transformar magnitudes físicas o químicas, llamadas variables de instrumentación, en magnitudes eléctricas.

Las variables de instrumentación dependen del tipo de sensor y pueden ser por ejemplo: temperatura, intensidad lumínica, distancia, aceleración, inclinación, desplazamiento, presión, fuerza, torsión, humedad, pH, etc. Una magnitud eléctrica obtenida puede ser una resistencia eléctrica (como en una RTD), una capacidad eléctrica (como en un sensor de humedad), una tensión eléctrica (como en un termopar), una corriente eléctrica (como un fototransistor), etc.

Los sensores pueden estar conectados a un computador para obtener ventajas como son el acceso a una base de datos, la toma de valores desde el sensor, etc.

a) Características de un sensor

Entre las características técnicas de un sensor destacan las siguientes:

- Rango de medida: dominio en la magnitud medida en el que puede aplicarse el sensor.
- Precisión: es el error de medida máximo esperado.
- Offset o desviación de cero: valor de la variable de salida cuando la variable de entrada es nula. Si el rango de medida no llega a valores nulos de la variable de entrada, habitualmente se establece otro punto de referencia para definir el offset.

- Linealidad o correlación lineal.
- Sensibilidad de un sensor: relación entre la variación de la magnitud de salida y la variación de la magnitud de entrada.
- Resolución: mínima variación de la magnitud de entrada que puede apreciarse a la salida.
- Rapidez de respuesta: puede ser un tiempo fijo o depender de cuánto varíe la magnitud a medir. Depende de la capacidad del sistema para seguir las variaciones de la magnitud de entrada.
- Derivas: son otras magnitudes, aparte de la medida como magnitud de entrada, que influyen en la variable de salida. Por ejemplo, pueden ser condiciones ambientales, como la humedad, la temperatura u otras como el envejecimiento (oxidación, desgaste, etc.) del sensor.
- Repetitividad: error esperado al repetir varias veces la misma medida.

Un sensor es un tipo de transductor que transforma la magnitud que se quiere medir o controlar, en otra, que facilita su medida..

b) Tipos de sensores

- **Ópticos:**

Detectan la presencia de una persona o de un objeto que interrumpen el haz de luz que le llega al sensor

Los principales sensores ópticos son las fotorresistencias, las LDR.

Son resistencias cuyo valor disminuyen con la luz, de forma que cuando reciben un haz de luz permiten el paso de la corriente eléctrica por el circuito de control. Cuando una persona o un obstáculo interrumpen el paso de la luz, la LDR

aumenta su resistencia e interrumpe el paso de corriente por el circuito de control.

En la figura 6 se grafica el rango de resistencia versus la intensidad luminosa, a una intensidad luminosa alta, el LDR tendrá una resitencia de aproximadamente 50 ohmios, mientras que a una falta de luminosidad, el LDR presenta una resistencia muy alta (Kohmios)

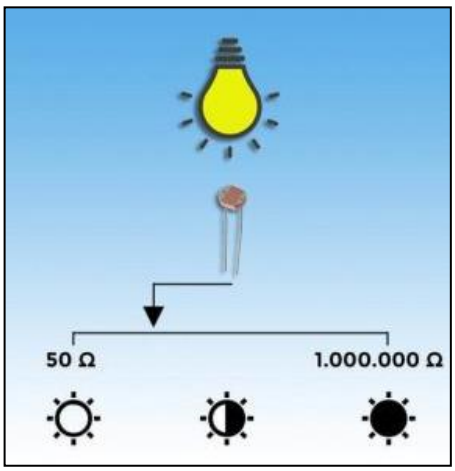

Figura 6: *Sensor LDR funcionando.*

Fuente*: programarfacil (programarfacil, 2018)*

- **Infrarrojos:**

Es un sensor con una fuente de luz (diodo emisor) y detector (fototransistor) integrados en un mismo encapsulado o de manera separada. La detección del objeto se consigue por la reflexión (o no) del haz infrarrojo sobre la superficie del objeto.

En la figura 7 se observa la gráfica de sensores infrarrojos tanto el emisor como el receptor, en este caso el haz de luz rebota en el objeto generando que el receptor detecte esta luz generando un cambio de estado en éste.

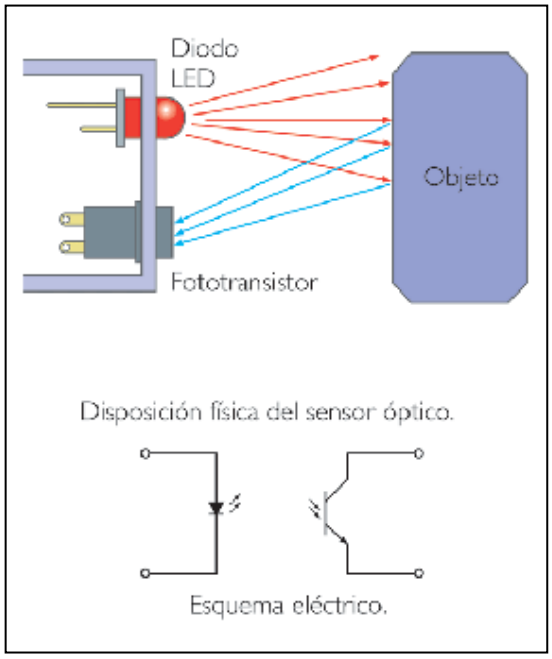

***Figura 7**: Sensor infrarrojo y su esquema eléctrico.*

***Fuente**: Recursostic.educacion (recursostic, 2017)*

- **Sensores de Contacto**

Se emplean para detectar el final del recorrido o la posición límite de componentes mecánicos. Por ejemplo: saber cuándo una puerta o una ventana que se abren automáticamente están ya completamente abiertas y por lo tanto el motor que las acciona debe pararse.

Los principales son los llamados fines de carrera (o finales de carrera). Se trata de un interruptor que consta de una pequeña pieza móvil y de una pieza fija que se llama NA, normalmente abierto, o NC, normalmente cerrado (Figura 8).

Figura 8: *Sensor de Contacto.*

Fuente*: Recursostic.educacion (recursostic, 2017)*

2.2.7. Higrómetro fc-28

El FC-28 es un sensor sencillo que mide la humedad del suelo por la variación de su conductividad. (Figura 9).

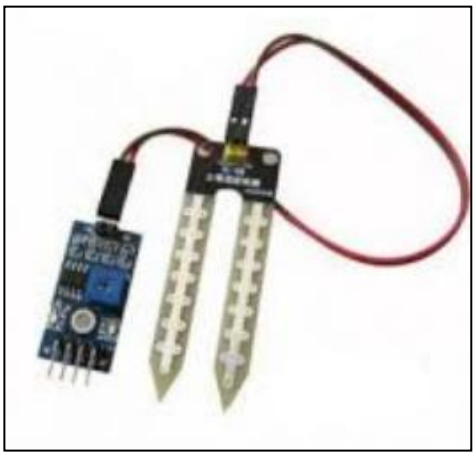

Figura 9: *Higrómetro fc-28.*

Fuente*: www.cronoselectronica.com (cronoselectronica, 2017)*

El FC-28 permite obtener la medición como valor analógico o como una salida digital, activada cuando la humedad supera un cierto umbral.

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

La salida digital dispara cuando el valor de humedad supera un cierto umbral, que ajustamos mediante el potenciómetro. Por tanto, obtendremos una señal LOW cuando el suelo no está húmedo, y HIGH cuando la humedad supera el valor de consigna. (Llamas, 2016)

2.2.8. Sensores de gas MQ4

Sensor del tipo electroquímico, su particularidad es que varía su resistencia cuando está expuesto a determinados gases, en el caso del proyecto será el gas propano; lo que permite este cambio de resistencia es un calentador interno que se encarga de aumentar la temperatura interna del dispositivo dando origen a una reacción del sensor cambiando el valor de su resistencia.

Este sensor se utilizará para la detección de fugas de gas en la maqueta, exactamente en el ambiente de la cocina. En la figura 10 se muestra el aspecto físico del Sensor MQ4.

Figura 10: *Sensor de gas MQ4.*

Fuente*: www.electromania.pe (electromania, 2017)*

2.2.9. Triacs

Un TRIAC o Triodo para Corriente Alterna es un dispositivo semiconductor, de la familia de los tiristores, podría decirse que el TRIAC es un interruptor capaz de conmutar la corriente alterna.

Este dispositivo en nuestro proyecto funcionará como un interruptor electrónico, que encenderá y apagara las luces de iluminación según las órdenes que envíe el Arduino.

En la figura 11 se muestra el símbolo, detalle de cada pata de conexión y el aspecto físico del Triac BT136 utilizado en el proyecto.

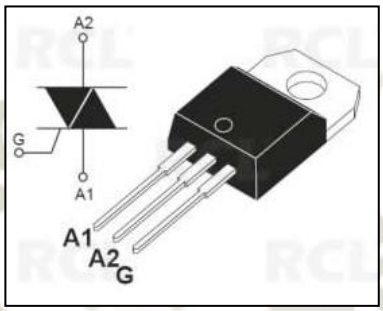

***Figura 11**: Aspecto físico de un Triac y su símbolo.*

***Fuente**: www. robotechshop.com (robotechshop, 2015)*

2.2.10. Optoacopladores

Un optoacoplador es un dispositivo de emisión y recepción.

Su funcionamiento consiste en un emisor y receptor, el emisor al estar activo emite luz infrarroja al receptor el cual se satura y permite la conducción de corriente.

En nuestro proyecto, estos dispositivos también funcionaran como interruptores que activaran los circuitos de alto voltaje. En la figura 12 se muestra el aspecto físico de los optoacopladores así como su esquema de conexión interna.

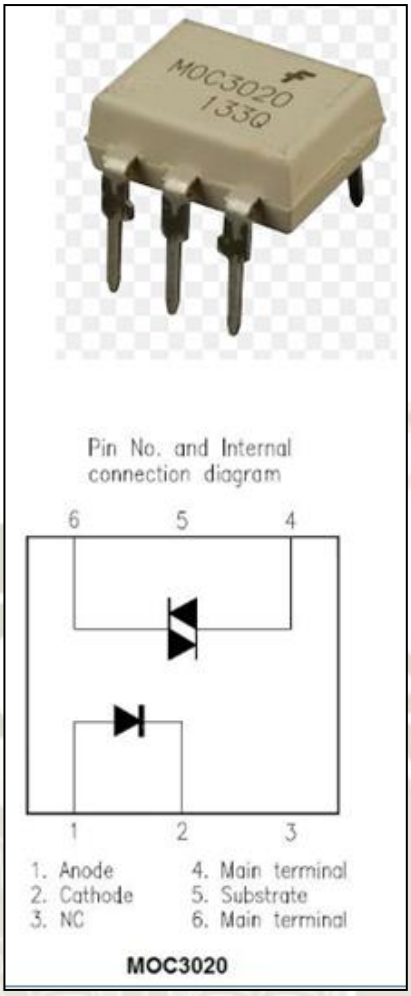

Figura 12: *Aspecto físico de un Optoacoplador y su símbolo.*

Fuente*: www.compic.es (compic, 2015)*

2.2.11. Servomotor

Un servomotor (también llamado servo) es un dispositivo similar a un motor de corriente continua que tiene la capacidad de ubicarse en cualquier posición dentro de su rango de operación, y mantenerse estable en dicha posición.

Internamente el eje del motor se acopla a una caja de engranajes, esto con la finalidad de aumentar el torque del motor y poder mantener una posición fija.

Los servomotores poseen tres cables, a diferencia de los motores comunes que sólo tienen dos. Estos tres cables casi siempre tienen los mismos colores, por lo que son fácilmente reconocibles.

Voltaje positivo	Tierra (ground)	Señal de control

Colores comunes de los cables de un servomotor.

Este tipo de motores no pueden funcionar sin un circuito de control debido a que es necesario para su funcionamiento una señal de control modulada, para esto se usa PWM (modulación de ancho de pulsos)

Estos dispositivos se utilizaran para abrir y cerrar puertas de habitaciones y puerta de garaje.

En la figura 13 se observa un servomotor con todos sus componentes internos, éste está compuesto por un motor eléctrico, engranajes que aumentan el torque del motor, y una placa controladora.

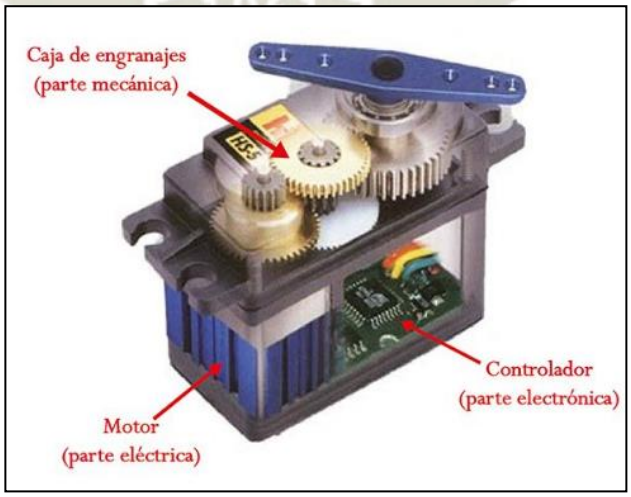

***Figura 13**: Servomotor al detalle.*

***Fuente**: www. panamahitek.com (panamahitek, 2014)*

2.2.12. Resistencias

La resistencia eléctrica es la oposición que presenta un conductor al paso de la corriente eléctrica. Su unidad es el Ohmio, que se representa con la letra griega omega (Ω). En la figura 14 se muestra el aspecto físico de una resistencia común de ¼ de Watt.

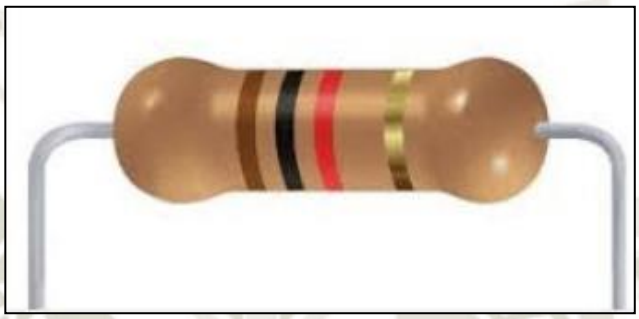

***Figura 14**: Aspecto físico de una resistencia.*

***Fuente**: ingenieriaelectronica.org (ingenieriaelectronica, 2014)*

En la figura 15 se pueden distinguir tres diferentes símbolos usados para las resistencias o resistores: el primero corresponde a resistencias eléctricas simples; el segundo a resistencias eléctricas variables (temperatura, luz, presión, humedad, etc.); mientras que el tercero representa resistores variables de tipo potenciómetro.

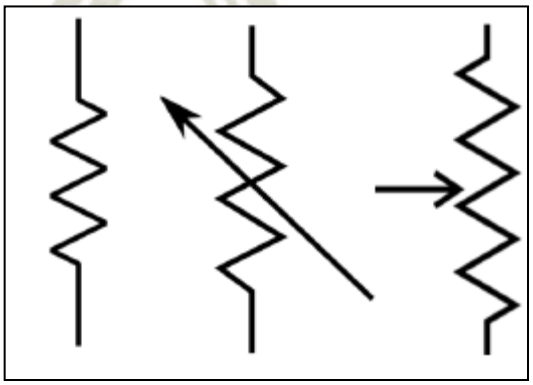

***Figura 15**: Simbología de una resistencia.*

***Fuente**: ingenieriaelectronica.org (ingenieriaelectronica, 2014)*

CAPÍTULO 3: PLAN DE IMPLANTACIÓN DEL PROYECTO.

3.1. Definición del Proyecto.

3.1.1. Aspectos Técnicos

El software consiste en la implementación de un servidor Web IIS el cual alojará una aplicación Web desarrollada en C# (Visual Studio 2013 Community), esta aplicación es del tipo responsiva, que instancia un puerto serie USB, el cual permite la comunicación con la placa Arduino Mega que recibirá las instrucciones de la aplicación Web y en consecuencia pondrá en funcionamiento los diferentes componentes electrónicos que permitirán por ejemplo abrir o cerrar puertas o garaje, encender o apagar luces, controlar equipos o electrodomésticos que usen control remoto, detección de fugas de gas, abrir o cerrar ventanas, etc.

3.1.2. Aspectos Económicos

Para el presente proyecto se utilizará los siguientes materiales y equipamiento:

DESCRIPCION	PRECIO EN SOLES
EQUIPO DE COMPUTO PORTATIL	1000
PLACA ARDUINO MEGA	50
OPTOACOPLADORES	10
TRIACS	10
PLACA DE BAQUELITA	10
SENSORES INFRAROJOS	5
RELAYS Y MINI DIP SWITCH	10
LEDS	5
TRANSFORMADOR DE CORRIENTE ALTERNA	15
SENSOR DE GAS	20

CABLES Y CONECTORES	20
SERVOMOTORES	50
TOTAL	S/1255.00

***Tabla 1**: Costo del proyecto*

***Fuente**: Elaboración propia*

3.1.3. Aspectos Comerciales

El presente proyecto no fue concebido con fines de lucro, fue pensado en dar ayuda a personas discapacitadas, contribuir con su bienestar, generar un proyecto social.

Dado que se utiliza hardware y software libre, el costo por licencias es cero, los circuitos, sensores y demás son de bajo costo y fácilmente pueden adaptarse para su funcionamiento en producción.

DESCRIPCIÓN DEL SISTEMA

El prototipo consta de 2 partes, una parte puramente electrónica y otra parte software:

3.2. Parte electrónica

La parte electrónica consiste de los siguientes circuitos:

3.2.1. Circuito de alta tensión

Este circuito permite manejar voltajes y corrientes mayores (220 V alterna, hasta 20 A) que permitirán controlar el encendido del ventilador y de un foco. El circuito se describe a continuación:

Según se detalla en la figura 16, el pin del Arduino a utilizar para encender el foco del baño será el número 22 y para encender el ventilador el pin 23, estos cambiaran de estado de 1 a 0 de acuerdo a las órdenes que se envíen por el entorno Web, si se

envía 1 (5v) este voltaje entra al circuito el cual es regulado por una resistencia de 470 ohmios, luego pasara por las patas 1 y 2 del optoacoplador MOC 3011; estas patas interconectan a un diodo led el cual se encenderá, este a su vez al emitir su luz, activara al fototransistor interno excitando su base la cual a su vez excitara a la compuerta del TRIAC BTA16, el cual en sus patas A1 y A2 están conectados en paralelo a los 220 v alternos de la toma de electricidad, que a su vez se conecta en serie al foco o al ventilador; dado que se envió un 1 lógico(5V) por el puerto correspondiente, el foco o ventilador se encenderá y en el ambiente Web se observa que el icono del foco cambia de apagado a encendido, en caso se presione nuevamente el icono del foco encendido en el ambiente Web, este enviará un 0 lógico el cual dejara de encender el led interno del optoacoplador y por ende ya no excitara al fototransistor y el circuito se quedara abierto apagándose el foco o el ventilador.

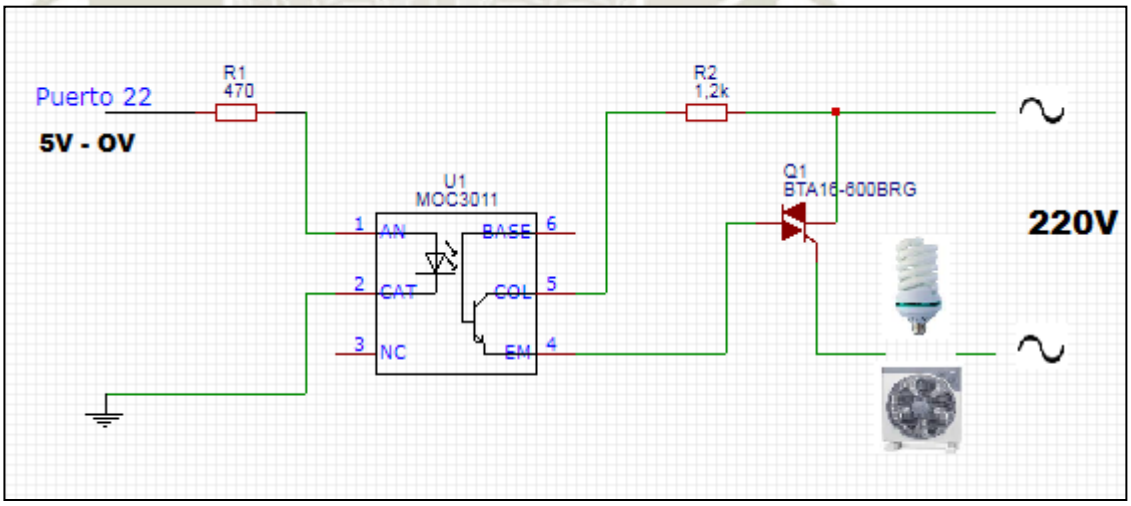

***Figura 16**: Circuito de alta tensión.*

***Fuente**: Elaboración propia*

Los circuitos de alta tensión implementados son 2, el triac que es el que actúa como interruptor o switch de encendido apagado, maneja altos voltajes y corrientes; a

continuación se describen las características principales del triac BTA16 usado en este proyecto.

- Corriente de Encendido Medio Nominal 16mA
- Corriente Máxima de Disparo de Puerta 100mA
- Tensión Inversa de Pico Repetitiva 600V
- Tensión Máxima de Disparo de Puerta 1.3V
- Corriente Máxima de salida 16A
- Temperatura Máxima de Funcionamiento +125 °C
- Temperatura de Funcionamiento Mínima -40 °C

3.2.2. Circuito de emisor receptor infrarrojo

Sensor de corte de haz de luz (usado como alarma); este circuito permite la detección de intrusión en el hogar, funciona a base de un diodo led emisor de luz infrarroja y un fotodiodo que detecta esta luz, estos están ubicados en la ventana del segundo piso de la casa.

Este circuito tiene 2 estados:

a) Estado de funcionamiento normal: Este estado se refiere al funcionamiento diario y sin interrupción del emisor y receptor, el emisor envía señal luminosa a un fotodiodo que también se encuentra en la ventana del segundo piso, estos 2 dispositivos están alineados uno frente a otro, este circuito es alimentado permanentemente para que así exista una continua emisión y recepción de señal infrarroja.

Debido a que el fotodiodo está recibiendo señal del emisor, este conduce corriente y se cierra el circuito; en los terminales que van al Arduino se tiene 0

voltios, esto quiere decir que el haz de luz infrarroja no ha sido interrumpido y por ende no hay intrusión en el hogar.

b) **Estado de activación-interrupción de haz infrarrojo – intrusión**: En este caso el haz de luz emitido por el emisor es interrumpido por algún cuerpo extraño lo cual genera que el fototransistor no conduzca electricidad, lo que generara la conducción del voltaje por el resto del circuito, la señal ingresara por la pata 2 del circuito integrado amplificador LM741 el cual amplificara la señal recibida en una escala de 100 veces más (esto debido a la conexión de una resistencia de 100k en la pata 2 y 6 de este integrado.

Esta señal amplificada será de apenas 2 voltios, pero para que se pueda enviar a Arduino y que esta active o desactive un pin, se debe de contar con al menos 5 voltios (1 lógico), por ese motivo es que se está usando el circuito de amplificación de voltaje adicional conformado por un transistor BC5478 el cual se saturara al momento que se reciban los 2 voltios por su base lo que generara un amplificación de los 2 voltios a 5 voltios continuos, este voltaje nos indica que se activó la detección de intrusos, esta señal se enviara al pin 10 del Arduino, esta entrada será validada por el sistema el cual encenderá una sirena y un led bicolor, indicativo de que se ha producido una intrusión en la casa.

En la figura 17 se grafica el detalle del circuito de detección de intrusos.

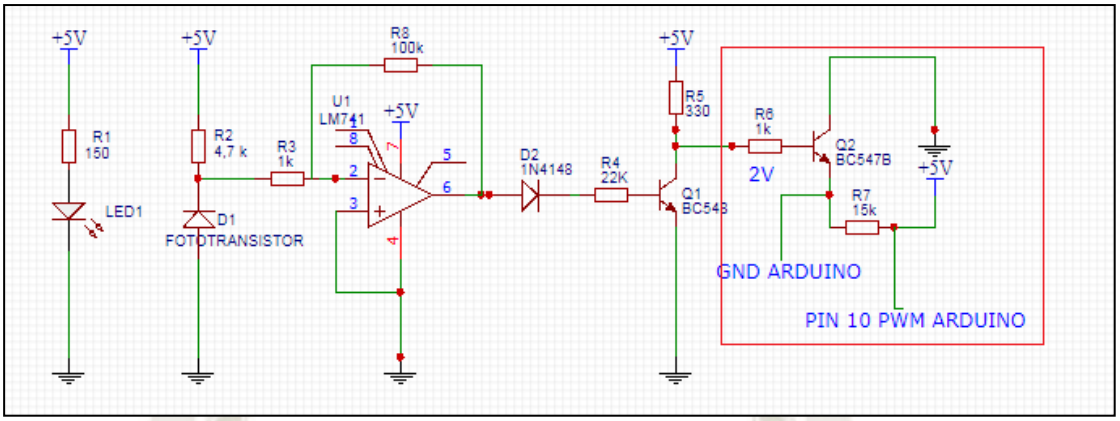

Figura 17: *Circuito sensor de corte de haz de infrarrojos.*

Fuente*: Elaboración propia*

3.2.3. Circuito de emisor de infrarrojos – controlar TV

Funcionamiento de un control remoto

Según se describe en LG (2017), los controles remotos funcionan a través de ondas de luces de baja frecuencia que son decodificadas por los aparatos. La onda es muy baja, por lo que no es perceptible para las personas.

Cuando se aprieta un botón del control, el botón activa un procesador, el cual a su vez encenderá una luz (diodo infrarrojo en la parte delantera del control), emitiendo la señal infrarroja hacia tu televisor.

El diodo emite una onda de luz infrarroja que, a pesar de ser invisible para el ojo humano, atraviesa el espacio y llega hasta el televisor. En la Figura 18 se observa una gráfica que detalla los diferentes rangos del espectro de luz existente La longitud de onda de la luz infrarroja va de 750 nanómetros a más.

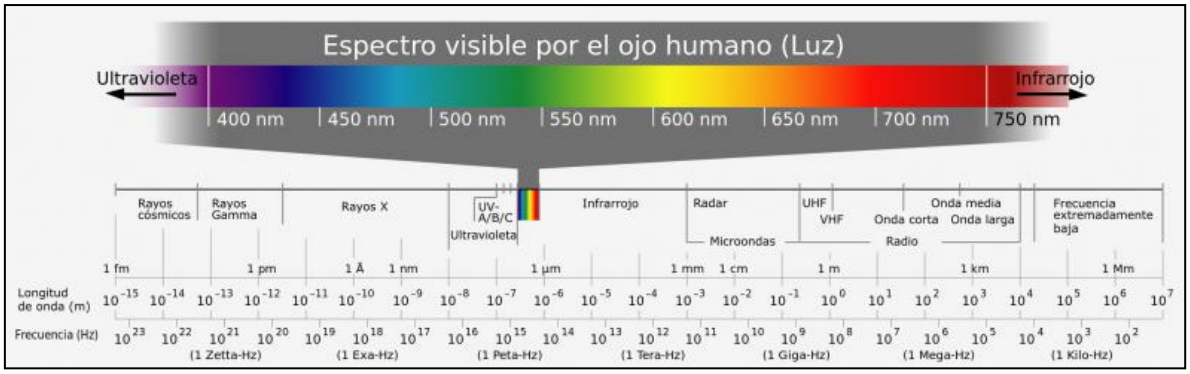

Figura 18: *Espectro de luz.*

Fuente*: www.vix.com (vix, 2017)*

El dispositivo a controlar recibe la onda de luz y la decodifica de tal modo que ejecuta la acción para la que estaba programado el botón del control.

El circuito de infrarrojos usado en nuestro Arduino nos permitirá controlar cualquier equipo o aparato que use un control remoto (en este caso se controlará un televisor), este circuito consta de 2 diodos led infrarrojos los cuales se conectan al pin número 9 del Arduino Mega, el uso de 2 led infrarrojos permite mayor radio de acción para el control del televisor (figura 19).

El entorno de la aplicación simula un control remoto el cual se detallara más adelante, esta aplicación enviara códigos hexadecimales que no son más que diferentes valores que son traducidos por el Arduino como pequeños pulsos de luz.

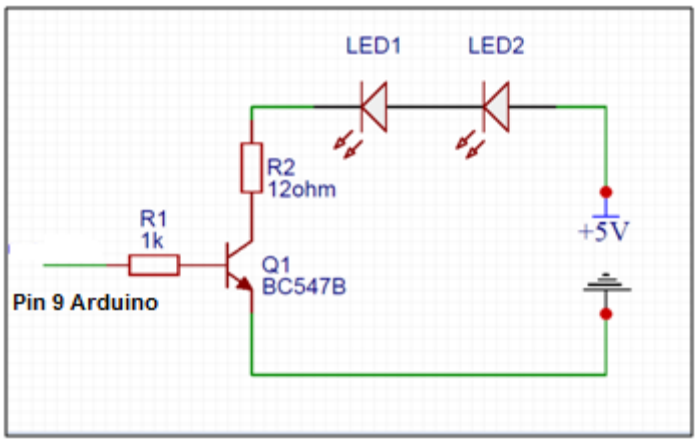

Figura 19: *Circuito emisión infrarrojos.*

Fuente*: Elaboración propia.*

3.2.4. Servo motores

El presente proyecto usa 2 servomotores, uno ubicado en la puerta del baño del primer piso y el otro usado para controlar la puerta de ingreso al ascensor.

El pin usado para controlar la puerta del cuarto de baño es el número 8 y para controlar la puerta del ascensor se utiliza el pin 10 de la sección PWM del Arduino en ambos casos, estos serán controlados usando valores de 0 grados para cuando estén cerradas las puertas, y de 90 grados cuando estén abiertas.

En la figura 20, se muestra la conexión para el servo motor que controlara la puerta del baño, como se observa el cable de datos del servo motor está conectado al pin 8 del Arduino mientras que el cable negro va a tierra y el rojo a 5 voltios de una fuente externa.

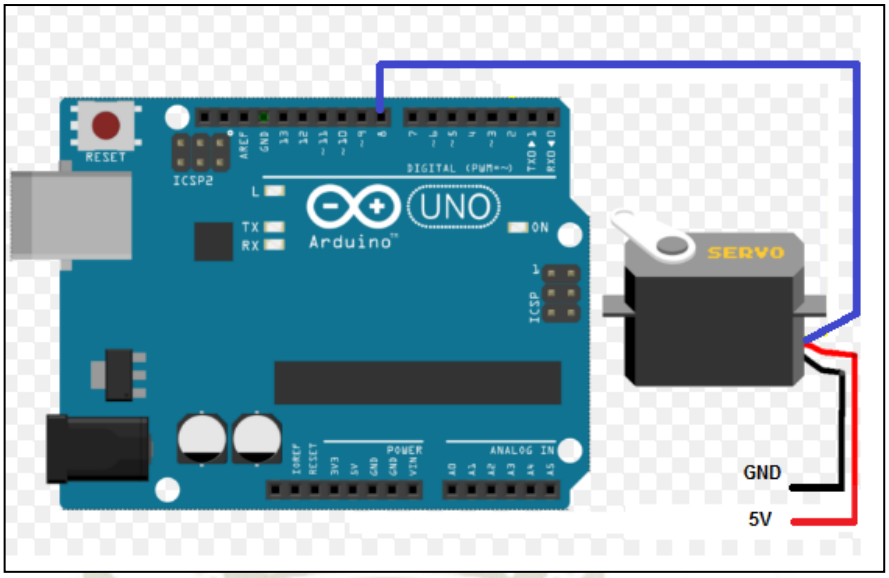

***Figura 20**: Servo motor Baño conectado a pin 8 de Arduino.*

***Fuente**: Elaboración propia.*

Lo mismo para el circuito que controla la puerta al ascensor (figura 21), el cable de datos del servo motor va conectado al pin 10 del Arduino mientras que el cable negro va a tierra y el rojo a 5 voltios de una fuente externa.

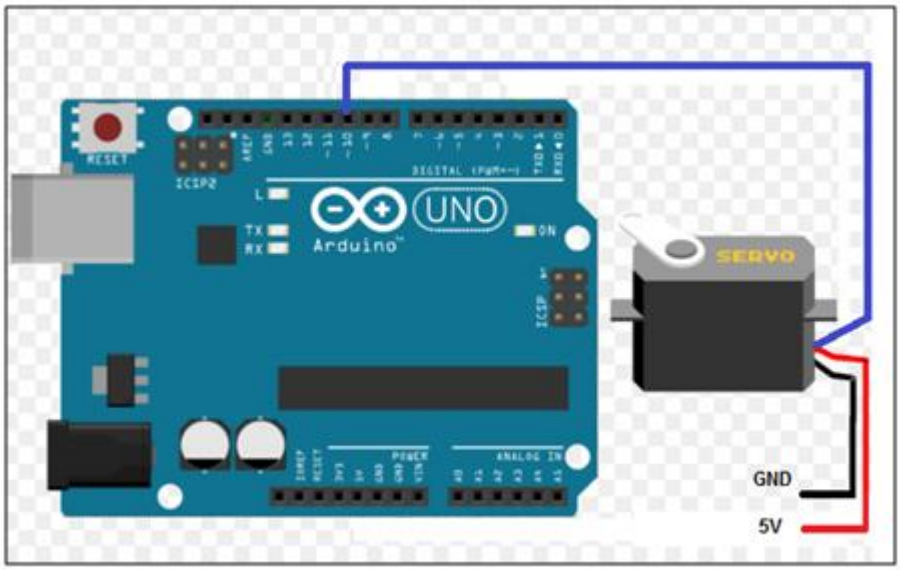

***Figura 21**: Servo motor Ascensor conectado a pin 10 de Arduino.*

***Fuente**: Elaboración propia.*

3.2.5. Control de giro de motores

El proyecto constara de 3 motores según se describe:

- 1 motor para subir/bajar puerta de garaje: para el control de este motor se usara los pines digitales 32 y 33.
- 1 motor para abrir/cerrar puerta de patio: para el control de este motor se usara los pines digitales 34 y 35.
- 1 motor para subir/bajar ascensor: para el control de este motor se usara los pines digitales 36 y 37.

El circuito integrado usado para el control de giro de estos motores es el L298 (Figura 22).

El L298N es un dispositivo que permite controlar el sentido de giro de funcionamiento de 2 motores a una corriente de salida por canal de hasta 2A, este control de giro se logra enviando voltajes de 0 o 5v por las entradas input 1 e input2 para el primer motor, input3 e input 4 para el segundo motor. (smching, 2015)

En la figura 22 se observa el detalle de los pines del integrado L298, cada integrado puede manejar 2 motores, los pines usados para el motor 1 (garaje) son:

Entradas digitales: pin 5 y 7
Salidas analógicas: pines 2 y 3
Los pines usados para el motor de puerta patio son:
Entradas digitales: pines 10 y 12
Salidas analógicas: pines 13 y 14

Para el control del motor de ascensor se utilizara el segundo integrado L298 los pines:

Entradas digitales: pin 5 y 7

Salidas analógicas: pines 2 y 3

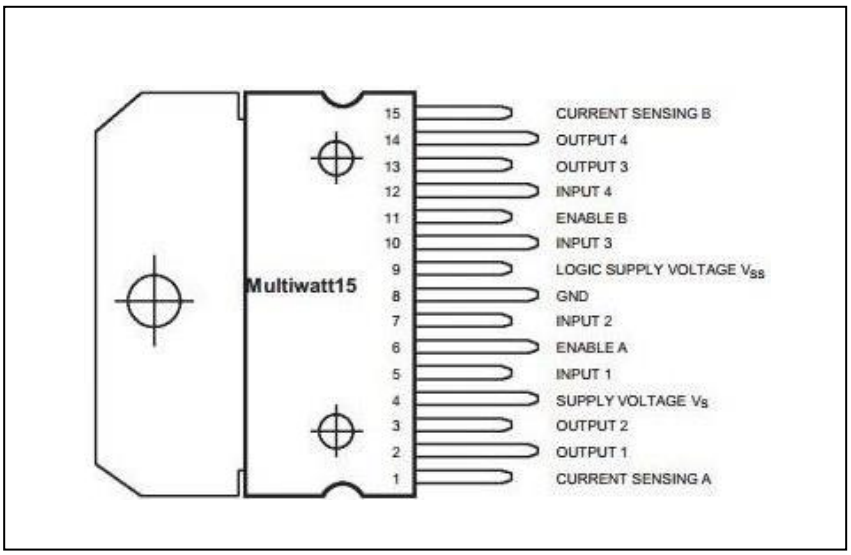

***Figura 22**: Detalle de pines de L298.*

***Fuente**: www.instructables.com(instructables,2015)*

En la figura 23 se detalla el esquema de conexión de los 2 motores (M1 y M2) y los pines utilizados del Arduino

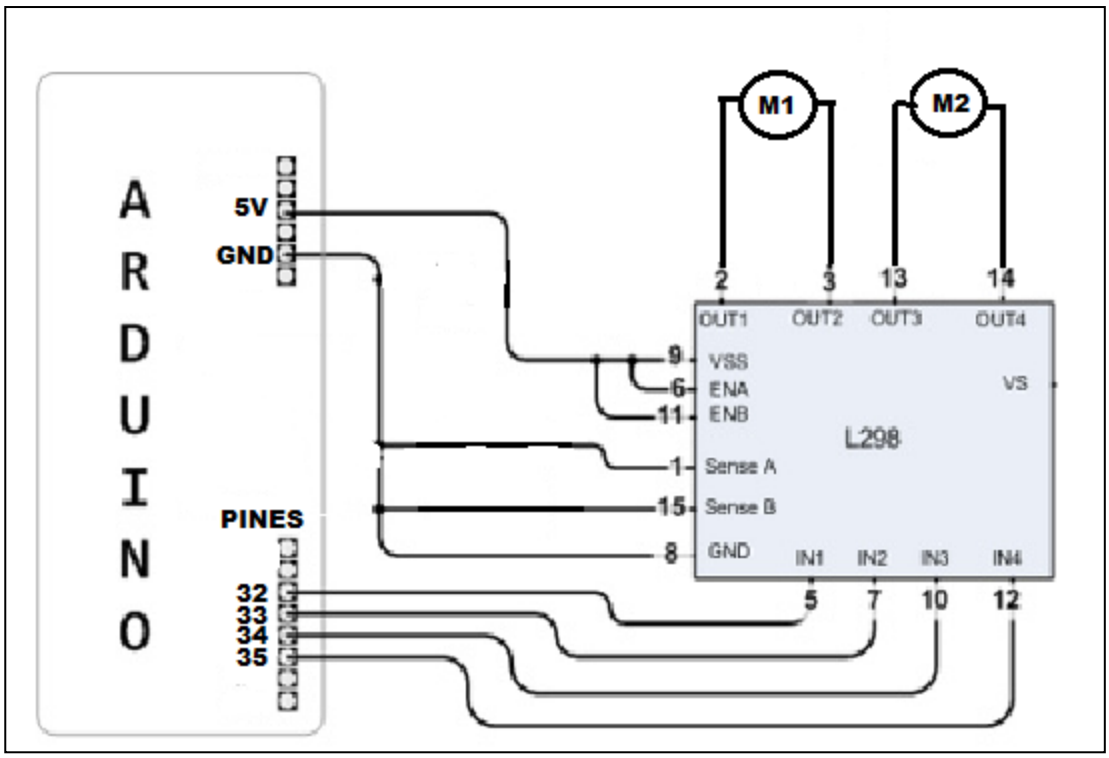

Figura 23: *Conexión de L298 a Arduino.*

Fuente*: Elaboración propia*

3.2.6. Circuito detector de gas propano

El presente proyecto está orientado a ayudar a personas en silla de ruedas, pero también puede servir para personas que puedan sufrir algún tipo enfermedad mental como el alzhéimer; estas personas puede sufrir dentro del hogar accidentes debido al olvido, como por ejemplo el olvido de cerrar una hornilla de gas. Para poder dar una posibilidad de protección a la persona, el proyecto contara con un sensor de gas MQ04 el cual será conectado al pin 7 PWM del Arduino, el cual detectara cualquier fuga de gas que se genere en la cocina, encendiendo un led de color rojo si fuera el caso de fuga, lo que ocasionara que la persona revise la llave de gas, evitando una posible explosión o envenenamiento por intoxicación con este combustible.

En la figura 24 se detalla el circuito sensor de gas utilizado en el proyecto, como se puede observar el cable de datos del sensor MQ04 va conectado al pin 7 del Arduino, el cable rojo va conectado a 5v de fuente externa y el negro a tierra..

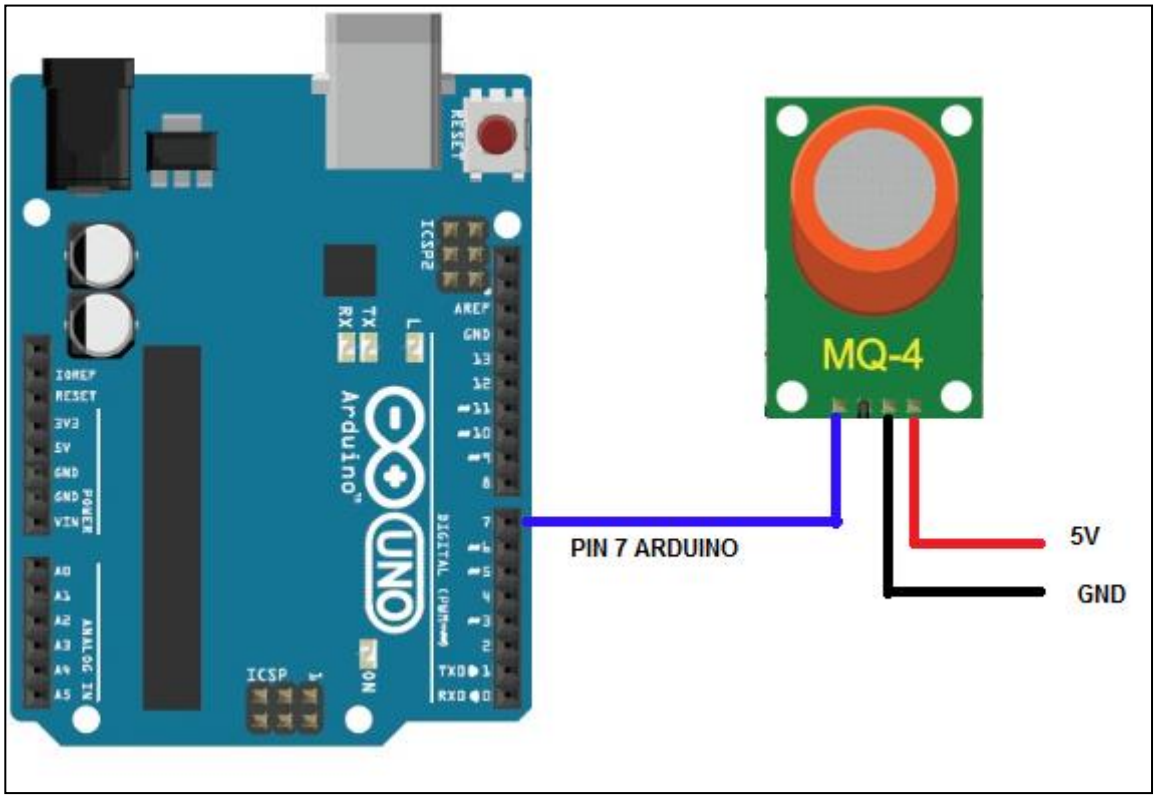

***Figura 24**: Sensor MQ4 conectado a Arduino.*

***Fuente**: Elaboración propia*

3.2.7. Riego controlado

El proyecto también permite el control del riego de jardines dando mayor autonomía a la persona discapacitada. El circuito consta de un sensor "Higrómetro fc-28", el cual se describió páginas arriba.

Este sensor va conectado al pin A0(entrada analógica) del Arduino el cual detectara si el suelo esta húmedo o seco, si fuese el caso que este seco, se habilitará

automáticamente una bomba la cual surtirá de agua a la tierra; cuando el suelo este suficientemente húmedo, el sensor detectará ésto y parará automáticamente el funcionamiento de la bomba.

En la figura 25 se detalla el circuito sensor de humedad, tal como se ve, hay un cable que va a tierra y otro a 5 voltios de alimentación, el tercer cable que es el de datos es el que va conectado al pin de entrada analógico A0 del Arduino.

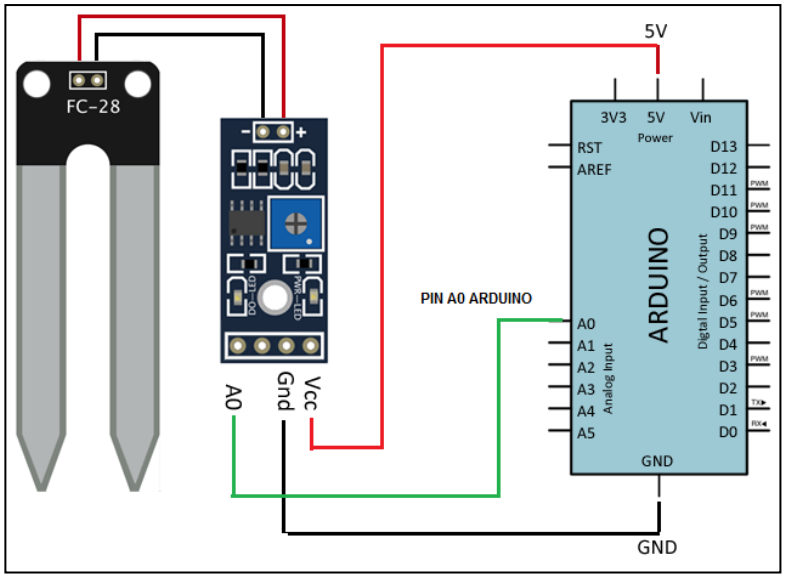

***Figura 25**: Higrómetro conectado a Arduino.*

***Fuente**: Elaboración propia*

3.2.8. Circuitos adicionales

Adicionalmente se tiene conectado al Arduino Mega 8 luces LED que simbolizan las luces de la casa, estas pueden ser reemplazadas sin ningún problema por circuitos de alta tensión como los usados para controlar la luz del cuarto de baño o el ventilador.

Estos Led son demostrativos, tienen como fin graficar y dar una idea más concreta sobre las posibilidades que se tiene en la conexión, control y distribución de iluminación.

Tal como se detalla en la figura 26, los pines usados son:

- Pin 24 de salida digital: este pin se usara para encender/apagar el led que simboliza el calefactor de la sala.
- Pin 25 de salida digital: este pin se usara para encender/apagar el led que simboliza la lámpara 1 de la sala.
- Pin 26 de salida digital: este pin se usara para encender/apagar el led que simboliza la lámpara 2 de la sala
- Pin 27 de salida digital: este pin se usara para encender/apagar el led que simboliza el foco 1 del patio.
- Pin 28 de salida digital: este pin se usara para encender/apagar el led que simboliza el foco 2 del patio.
- Pin 29 de salida digital: este pin se usara para encender/apagar la sirena que indicara la intrusión de personas a la casa.
- Pin 30 de salida digital: este pin se usara para encender/apagar el led que simboliza la luz externa de la puerta de ingreso principal.
- Pin 31 de salida digital: este pin se usara para encender/apagar el led que simboliza la luz externa de la puerta de garaje de la calle.

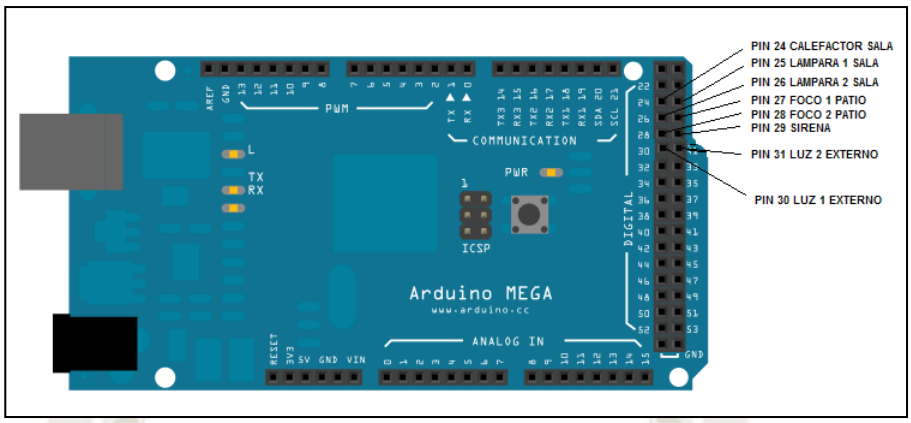

***Figura 26**: Detalle de pines usados para control de luces.*

***Fuente**: Elaboración propia*

3.3. Parte software

La aplicación Web utilizada en el presente proyecto está desarrollada en Visual Studio 2013 Comunnity Server, la arquitectura del proyecto está basada en el patrón MVC.

Esta aplicación consta de una página principal de login la cual conecta a una base de datos implementada en Microsoft SQL server, esta valida a los usuarios para el ingreso a la interfaz principal.

Inicialmente esta base de datos tiene una sola tabla la cual almacena a los usuarios y sus contraseñas, ésta tabla se describe a continuación:

TABLA USUARIOS		
PK	UsuarioId	int
	Nombre	varchar (10)
	Password	varchar (10)
	Email	varchar (50)
	EsActivo	bit

***Tabla 2**: Tabla usuarios*

***Fuente**: Elaboración propia*

La Web es de tipo responsiva, esto quiere decir que la aplicación como los gráficos de maquetas de cada piso y habitación se adaptan a la resolución de la pantalla del dispositivo en el que se esté ejecutando, esto se logra gracias al uso de librerías de bootstrap.

En la figura 27 se muestra la interfaz inicial de la aplicación, la página Web principal de login:

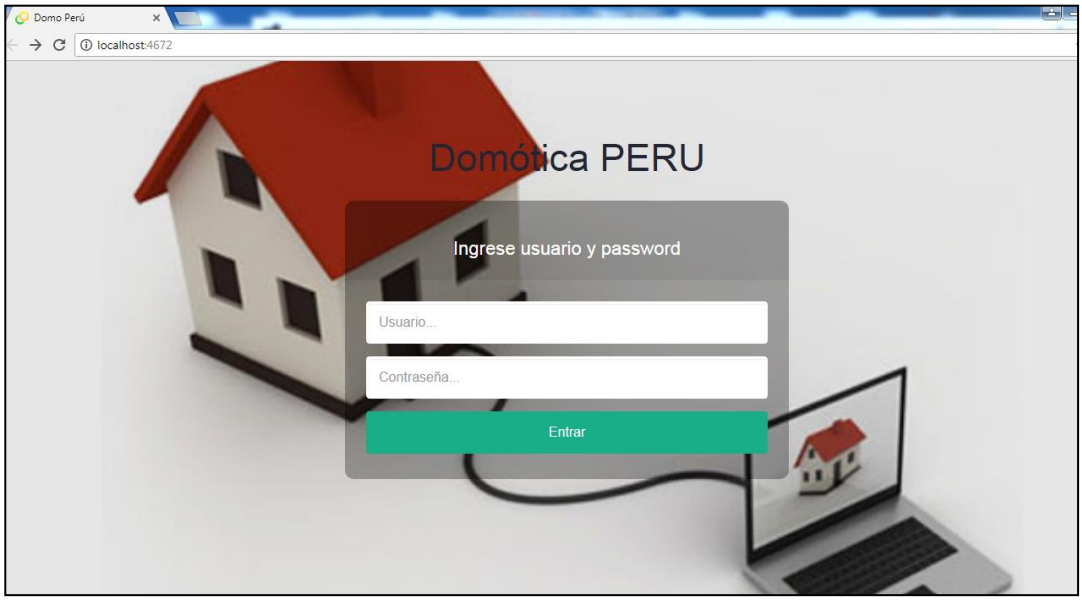

***Figura 27**: Pagina Web inicial del proyecto*

***Fuente**: Elaboración propia*

Si se ingresa un usuario o password errado el sistema validará esto y mostrará el mensaje “Usuario o password incorrectos” según se puede observar en la figura 28.

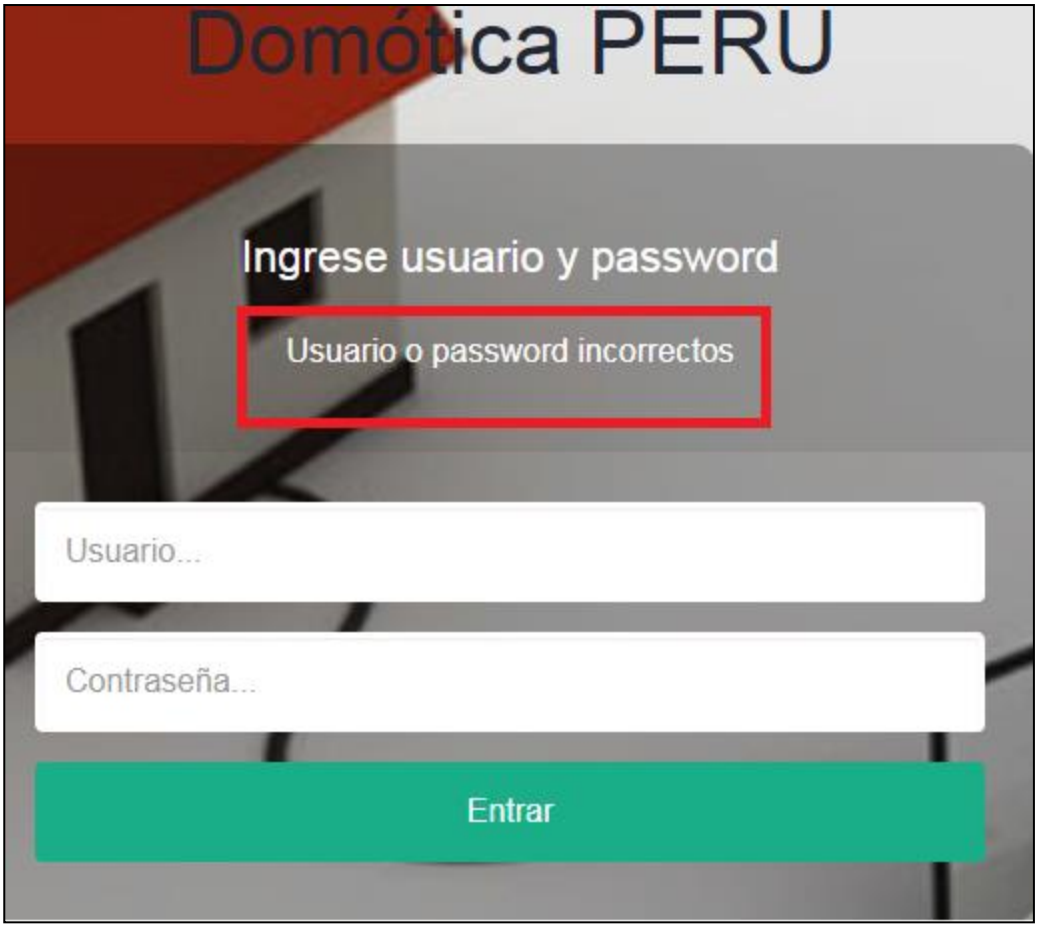

Figura 28: *Validación de datos*

Fuente*: Elaboración propia*

Una vez autenticado correctamente con un usuario existente en la base de datos, el sistema redirecciona a la página "index", Figura 29.

***Figura 29**: Acceso a los diferentes pisos*

***Fuente**: Elaboración propia*

En la parte superior izquierda si se hace click en el botón collapse se mostrara un menú del tipo side bar el cual muestra las opciones de la figura 30:

***Figura 30**: Detalle del Sidebar*

***Fuente**: Elaboración propia*

- **Opción PISO**: haciendo click aquí, se muestran las opciones Piso 1 y Piso 2, cada una nos llevará a controlar tanto las habitaciones del piso 1 como del piso 2.
- **Opción PERIMETRO**: Haciendo click en esta opción, el sistema redirecciona a otra página en la cual podremos controlar las luces exteriores de la casa.
- **Opción RIEGO:** Haciendo click aquí, el sistema nos redireccionara a la página en la cual podremos controlar el riego.
- **Opción Alarma:** Haciendo click aquí, el sistema nos redireccionara a la página en la cual podremos controlar la alarma.

A continuación se describe la opción Piso 1:

3.3.1. Piso 1

Al momento de hacer click en PISO 1, el sistema cargará la página en la cual se muestra un plano de todo el piso 1, con todos sus ambientes (figura 31)

***Figura 31**: Plano Primer Piso*

***Fuente**: Elaboración propia*

Al hacer click en cada cuarto, nos redireccionará a la página correspondiente en la cual se muestran todos los dispositivos que pueden controlarse.

3.3.2. Sala

Al hacer click en este cuarto nos cargara la página que se muestra en la figura 32.

***Figura 32**: Plano de Sala*

***Fuente**: Elaboración propia*

Cabe indicar que las imágenes que simbolizan los dispositivos cambian de estado (color o forma) al momento que son activados /desactivados, así mismo también se actualiza su estado automáticamente al momento que se ingresa o se actualiza la página, esto gracias al uso de javascript.

A continuación se detalla los dispositivos que pueden ser controlados:

a) Calefactor

Al presionar esta imagen, se encenderá un calefactor, en el proyecto se simboliza este calefactor con una maqueta que enciende un led de color rojo. Al hacer click en el icono del calefactor, cambiará de color (rojo) indicativo de que el calefactor esta encendido, si se vuelve a dar click el icono se vuelve de color blanco que simboliza el apagado del calefactor. (Figura 33).

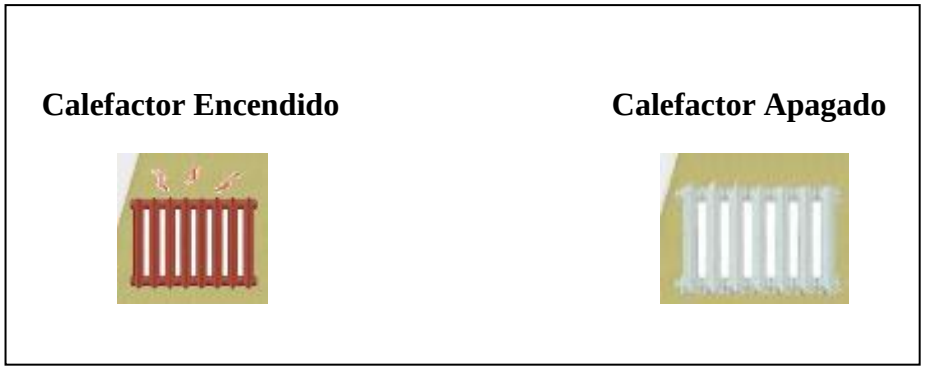

***Figura 33**: Estados del calefactor*

***Fuente**: Elaboración propia*

La señal que controla este encendido apagado del calefactor puede utilizarse perfectamente para encender un calefactor de verdad, esto se lograría con un circuito de manejo de potencia que no es muy costoso.

Los calefactores normalmente consumen entre 1000 watts a 1500 watts, esto se podría controlar con el triac Q6040K7 que maneja entre 6 a 40 amperios esto equivale hasta 8800 watts.

b) Ventilador

Al presionar esta imagen se activara el pin 23 de salida digital del Arduino, la

cual activará el circuito de alta tensión, encendiendo el ventilador que funciona a 220 voltios.

Los 2 estados o formas que adquiere el icono de ventilador se detallan en la figura 34.

***Figura 34**: Estados del ventilador*

***Fuente**: Elaboración propia*

c) Ascensor

Ser una persona discapacitada trae consigo la imposibilidad de movilizarse de un piso a otro superior y viceversa, para dar solución a este problema, se propone la instalación de un ascensor que también será controlado por medio del Arduino, este tendrá una puerta que es controlada por un servo motor(el estado de esta puerta se detalla en la figura 35) y por el pin número 9 de la sección PWM del Arduino, que al momento de hacer click en el icono de la puerta, el servo motor moverá su eje en un ángulo de 90 grados logrando así aperturar la puerta, una vez que el discapacitado haya subido al ascensor, la puerta deberá de cerrarse, para lo cual el sistema envía el valor de 0 grados al servomotor, cerrando la puerta.

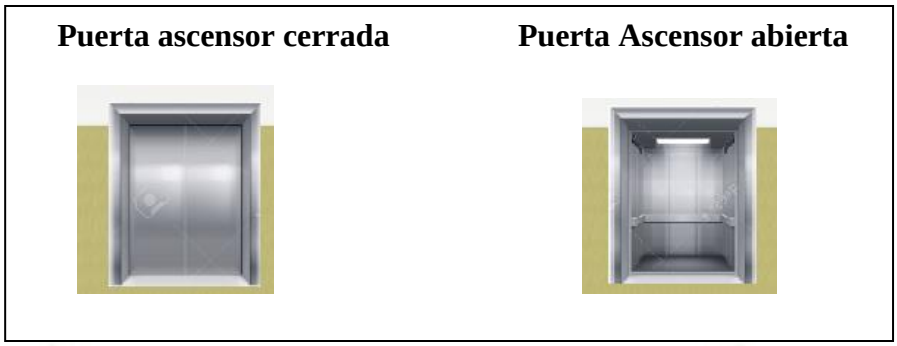

***Figura 35**: Estados de puerta de ascensor*

***Fuente**: Elaboración propia*

Para subir o bajar el ascensor, el discapacitado deberá de presionar el icono de flecha arriba ubicado en el cuarto de sala del primer piso, este icono cambiara de estado a "flecha abajo" el cual al presionarlo, el ascensor bajara al primer piso (figura 36).

El motor que sube/baja el ascensor, está controlado por los pines 36 y 37 de las salidas digitales del Arduino, los cuales envían señales de 0 y 1 (0v y 5v) al controlador L298.

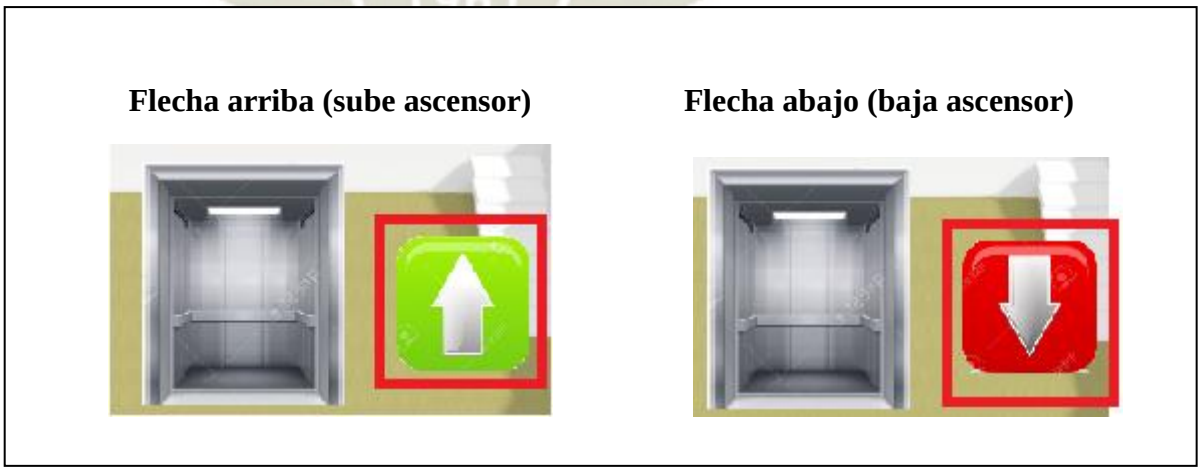

***Figura 36**: Estados de subir/bajar ascensor*

***Fuente**: Elaboración propia*

d) Control de TV

Al momento que se presiona el ícono del televisor, el sistema redirecciona a la página que nos muestra un control remoto en la cual podremos usarlo para controlar el televisor (figura 37).

Al momento que se presionan los diferentes botones del control, se activa el pin 3 de la sección PWM de Arduino, el cual según el valor programado para cada caso, el circuito de infrarrojos emitirá a la frecuencia adecuada señales que permitirán por ejemplo prender la tv, cambiar de canal, subir volumen, etc.

***Figura 37**: Pagina Web de control remoto de tv*

***Fuente**: Elaboración propia*

e) Control de luces

Al momento que se presiona el icono de control de luces, el sistema nos envía a la página en la cual aparecen las luces que se podrán controlar, estas luces son leds que representan a las luces reales.

Los pines del Arduino utilizados para encender estas luces son el pin 25 y 26 de las salidas digitales del Arduino.

3.3.3. Garaje

Al momento de hacer click en el cuarto de garaje, el sistema nos envía a la página donde se muestra la puerta de garaje, esta es controlada por los pines 32 y 33 de las salidas digitales del Arduino, al igual que los demás botones, cambia de estado según este abierta o cerrada esta puerta (figura 38).

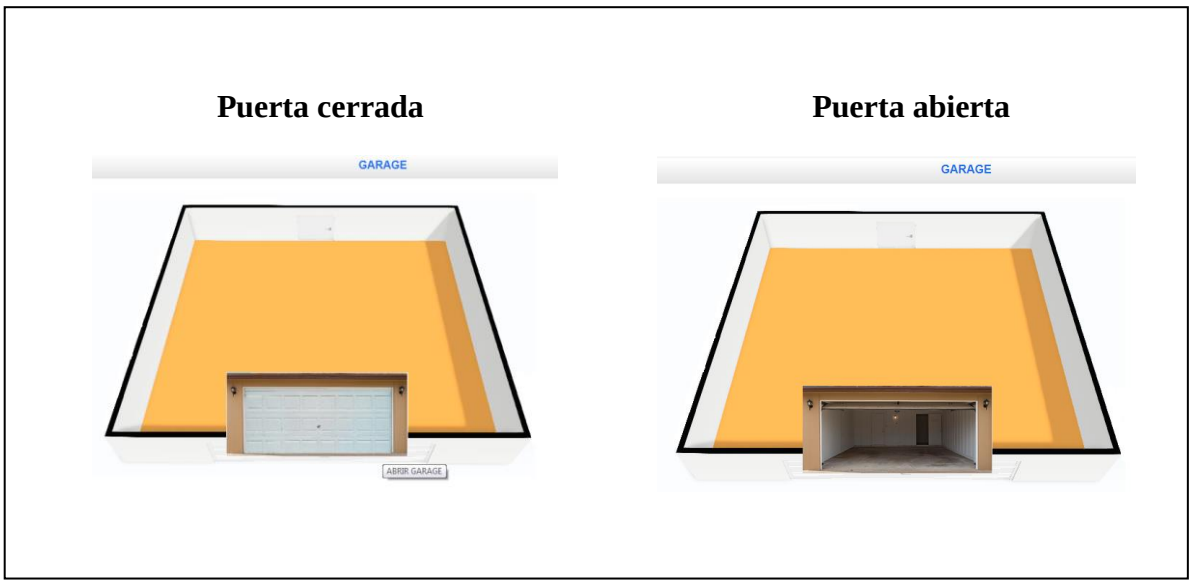

***Figura 38**: Estados de puerta de garaje*

***Fuente**: Elaboración propia*

3.3.4. Baño

El siguiente ambiente es el baño del primer piso, al momento que presionamos su icono, navegamos al plano de este cuarto (detallado en la figura 39), en él se puede controlar la puerta y la luz, de igual forma, si se presiona el foco éste se encenderá.

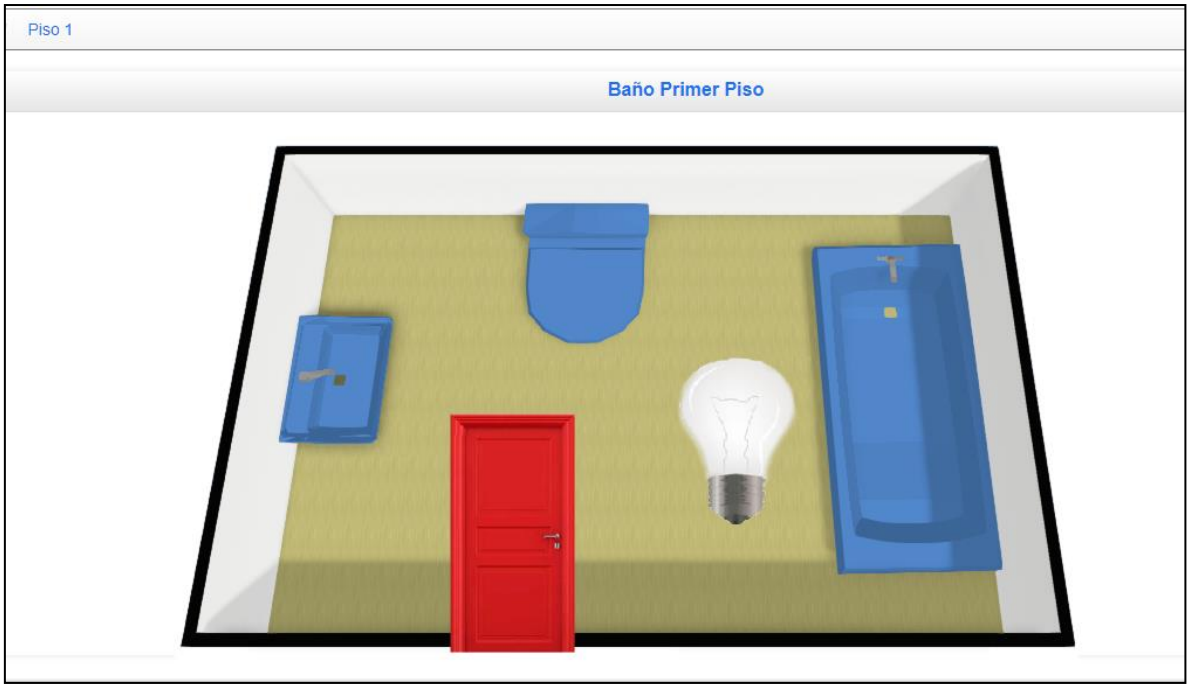

***Figura 39**: Cuarto de baño*

***Fuente**: Elaboración propia*

El ícono del foco también cambiará de color blanco a color amarillo que simboliza que está encendido tal como se muestra en la figura 40.

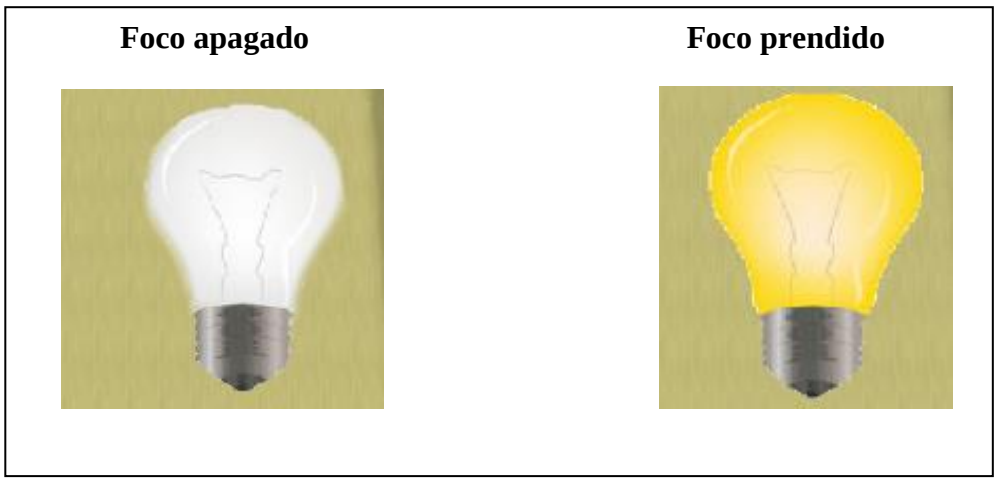

***Figura 40**: Estados del foco*

***Fuente**: Elaboración propia*

Para el caso de la puerta de ingreso al baño, sucederá algo similar; al hacer click en ésta, se accionara el pin 8 de la sección PWM del Arduino, haciendo girar al servo motor a 90 grados abriendo la puerta.

Presionando nuevamente el servo motor girará a 0 grados y cerrará la puerta.

El ícono de la puerta también cambiará de estado según se muestra en la figura 41.

***Figura 41**: Estado de puerta de baño*

***Fuente**: Elaboración propia*

3.3.5. Cocina

En el ambiente de la cocina se cuenta con un sensor detector de gas (MQ04 de Arduino); el icono representativo de este sensor cambiará a “Peligro” y se encenderá un led de color rojo, esto será indicativo de que existe una fuga de gas alertando al usuario para que controle esta fuga y así evitar accidentes mayores.

En la cocina también se cuenta con una puerta que da al patio, ésta es controlada por un motor el cual permite abrir y cerrarla, esta puerta está controlada por los pines 34 y 35 de las salidas digitales del Arduino; al igual que los anteriores casos, se mostrará el icono de una puerta cerrada que al hacerle click, activará el motor y permitirá abrir y cerrar la puerta de manera horizontal, cambiando el icono a una puerta abierta.

Como fin de carro se tienen 2 sensores de contacto que se encargan de abrir el circuito al momento que se activan para que el motor deje de girar.

El control del giro de motor se logra con el circuito integrado L298.

3.3.6. Patio

Para el patio se cuenta con el control de 2 luces las cuales son 2 leds de color blanco que son activadas por los pines 30 y 31 de las salidas digitales del Arduino.

Al igual que los otros casos, estos iconos cambiaran de estado de encendido/apagado al momento que se presionen.

3.3.7. Perímetro

En el menú Side Bar, existe la opción “Perímetro”, al hacer click se mostrará una gráfica del perímetro de la casa, en la cual se pueden encender los focos de la entrada principal y de la entrada al garaje.

Estas luces son leds de color blanco, pero que perfectamente pueden ser cambiados por focos de 220 voltios usando el circuito de triacs de alto voltaje.

El encendido de éstas luces son controladas por los pines 27 y 28 de la salida digital del Arduino.

3.3.8. Riego.

En el menú Sidebar también aparece la opción "Riego", al hacer click, el sistema nos llevará a la página de riego, en esta página se podrá activar la bomba de agua que activará el sistema de riego del patio, esta bomba está controlada por los pines 42 y 43 de la salida digital del Arduino, que a su vez envían señal digital a las entradas del circuito de control de motores L298.

Otra particularidad que tiene el sistema de riego es el uso de un sensor de humedad el cual al detectar un nivel bajo de humedad en la tierra, activa automáticamente la bomba de agua que regará la tierra hasta que el nivel de humedad programado en el Arduino sea alcanzado, momento en el que la bomba se detendrá.

3.4. Descripción del programa Arduino.

Como se indicó, Arduino utiliza el lenguaje de programación Procesing el cual es muy similar al C++, a continuación de describe los diferentes bloques del programa Arduino.

3.4.1. Inclusión de librerías

a) librería servo.h

Esta biblioteca permite el manejo de servomotores, la biblioteca Servo admite hasta 12 motores en la mayoría de las placas Arduino y 48 en el Arduino Mega.

A continuación se describen las principales funciones usadas en el proyecto:

- **attach(Pin)**

  Establece el pin indicado en la variable servo. Ej: *servo.attach(2);*

- **write(ángulo)**

  Envía la señal correspondiente al servo para que se ubique en el ángulo indicado, el parámetro "ángulo" es un valor entre 0 y 180°. Ej: *servo.write(45);*

b) librería irremote.h:

Esta biblioteca permite enviar y recibir códigos de control de infrarrojo.

IRremote es una de las librerías más usadas y completas para trabajar con protocolos de controles infrarrojos, tiene implementado varios protocolos de las marcas más conocidas como Sony, LG, Samsung, Sanyo, etc.

Para trabajar con esta librería se tiene que añadir en la cabecera del programa principal con el siguiente comando:

#include <Irremote.h>

3.4.2. Declaración de variables para control de tv

Según se observa en la figura 42, en este bloque de código se instancia una variable IRsend (definida en la librería IRremote.h) con la cual se enviará los códigos respectivos de cada canal.

Adicionalmente se declaran variables “unsigned int”, en ellas se almacenan los códigos correspondientes a cada canal o función del televisor

```
//variable control remoto infrarrojo
IRsend irsend;
unsigned int onoff[52] = {3800, 3950, 500, 1950, 500, 1950, 500, 1950,
500, 1950, 450,  1050, 450,  1000, 500,  1950, 500,  1000, 450, 2000,
450,  1000, 500,  1950, 500,  1000, 500,  1000, 450,  1000, 550,  950,
500,  1000, 450,  2000, 500,  1950, 450, 1000, 500,  1950, 500,  1000,
500,  1950, 500,  950, 500,  1950, 500};
unsigned int canal1[52] = {3800,  3950, 500,  1950, 500,  1950, 500,
1950, 500,  1950, 500,  950, 500,  1000, 500,  1950, 500,  1950, 500,
1000, 450,  1000, 500,  1000, 500,  1950, 500,  1000, 450,  1000, 500,
1000, 500,  1000, 500,  1950, 450,  1950, 500,  1000, 500,  1000, 500,
1950, 500,  1950, 450,  2000, 450,  1000, 500};
unsigned int canal2[52] = {3850,  3950, 500,  1950, 500,  1950, 500,
1950, 500,  1950, 500,  1000, 500,  950, 500,  1950, 500,  1950, 500,
1000, 500,  950, 500,  1950, 500,  1000, 500,  950, 500,  1000, 500,
1000, 500,  1000, 500,  1950, 450,  1950, 500,  1000, 500,  1000, 500,
1950, 500,  1950, 500,  950, 500,  2000, 450};
unsigned int canal3[52] = {3800,  4000, 500,  1900, 500,  1950, 550,
1900, 550,  1900, 500,  1000, 500,  950, 500,  1950, 500,  1950, 550,
950, 500,  1000, 500,  1900, 500,  1950, 550,  950, 550,  950, 500,
1000, 500,  950, 500,  1950, 500,  1950, 500,  950, 500,  1000, 500,
1950, 500,  1950, 500,  950, 550,  950, 550};
```

Figura 42: *Declaración de variables irsend*

Fuente*: Elaboración propia*

3.4.3. Declaración de variables de control del servo motor

Para que puedan funcionar los Servo motores se deben de instanciar los objetos del tipo Servo. En el bloque de código siguiente se instancia los objetos que manejaran a los servos, el objeto ServoBano corresponde al Servo motor que abre/cierra la puerta del baño mientras que el objeto ServoAsc corresponde al manejo de Servo motor que abre/cierra la puerta del ascensor.

Comando “Servo ServoBano”: instancia el objeto ServoBano del tipo “Servo” , permite el manejo del servomotor que abre/cierra la puerta del baño

Comando "Servo ServoAsc": instancia el objeto ServoAsc del tipo "Servo", permite el manejo del servomotor que abre/cierra la puerta del ascensor

Luego de instanciar los objetos Servo, se debe de inicializar el ángulo de giro de los servomotores a 0 grados (puertas cerradas), para esto se utilizan los comandos

ServoBano.write(0);

ServoAsc.write(0);

Variable para Sensor de gas: Para el control del sensor de gas MQ04 utilizaremos el pin 7; el siguiente comando define la variable pin_mq y le asigna el valor 7 que es el número de pin que se usará para controlar el sensor de gas:

Int pin_mq=7:

Las variables que se encargan del control de luces se detallan en figura 43, como se observa, se declaran las variables que controlarán la luz del baño, el ventilador, así como también la declaración de variables de control del calefactor, las lámparas de la sala, los focos del patio, el foco de la puerta principal y el foco del garaje.

```
//---------------Control de salidas-----------------

//Variable de alto voltaje
int luzBano = 22;   //pin control luz baño
int ventilador=23;  //pin control ventilador
//Variable led
int radiadorSala=24;
int lamp1 = 25;
int lamp2 = 26;
int focoPatio1=27;
int focoPatio2=28;
int focogas=29;
int focoEntrada=30;
int focoGarage=31;
```

***Figura 43**: Declaración de variables y pines de control*

***Fuente**: Elaboración propia*

Detalle de variables utilizadas para el control de los diferentes motores

```
//Variable control motor garage
int ptaGarage1=32;
int ptaGarage2=33;

//Variable control motor puerta patio
int ptaPatio1=34;
int ptaPatio2=35;

//Variable control motor ascensor
int ascen1=36;
int ascen2=37;

// Variables para el control del motor de riego
int riego1=42;
int riego2=43;
```

***Figura 44**: Declaración de variables de motores.*

***Fuente**: Elaboración propia*

3.4.4. Sección void setup( )

En la figura 45 se declara el tipo de funcionamiento de cada pin, si serán de salida o de entrada, en este caso todas las variables se declaran como salida “OUTPUT”

```
//configuracion del tipo de funcionamiento de cada pin
  pinMode(luzBano, OUTPUT);
  pinMode(ventilador, OUTPUT);
  pinMode(radiadorSala,OUTPUT);
  pinMode(lamp1, OUTPUT);
  pinMode(lamp2, OUTPUT);
  pinMode(focoPatio1, OUTPUT);
  pinMode(focoPatio2  , OUTPUT);
  pinMode(focogas,OUTPUT );
  pinMode(focoEntrada, OUTPUT);
  pinMode(focoGarage, OUTPUT);
  pinMode(ptaGarage1, OUTPUT);
  pinMode(ptaGarage2, OUTPUT);
  pinMode(ptaPatio1, OUTPUT);
  pinMode(ptaPatio2, OUTPUT);
  pinMode(ascen1, OUTPUT);
  pinMode(ascen2, OUTPUT);
  pinMode(pin_corte, OUTPUT);
  pinMode(puertaAsc,OUTPUT );
  pinMode(puertaBan,OUTPUT );
  pinMode(riego1, OUTPUT);
  pinMode(riego2, OUTPUT);
```

***Figura 45**: Definición del tipo de pin.*

***Fuente**: Elaboración propia*

3.4.5. Sección void loop ( ).

Aquí va el cuerpo principal del programa, donde se encuentran las ordenes y funciones que interactúan con los diferentes dispositivos conectados al Arduino. Esta sección es iterativa, quiere decir que siempre está en ejecución.

3.4.6. Control de sistema de riego.

Según muestra la figura 46, si el flag toma el valor de 1, el sistema de riego entrará en funcionamiento; en la variable humedad se almacena el resultado de leer el pin de riego, este valor es análogo y va de un rango de 0 a 1000, para valores pequeños será indicativo que el sensor esta húmedo mientras que para valores más altos será un indicativo que el sensor está seco.

En este caso tomamos un valor de 350 para adelante, si éste rango es alcanzado, seteará las variables de riego una a alto y la otra a bajo, lo que ocasionara el encendido de la bomba de agua, caso contrario las variables se pondrán en nivel bajo “LOW” apagando la bomba de agua.

```
//-----------------RIEGO--------------------
//Serial.println(flag);
if (flag==1)
{
  humedad=analogRead(riego);
  if(humedad > 350)
  {
    digitalWrite(riego1, HIGH);
    digitalWrite(riego2, LOW);

  }

  else
  {
     digitalWrite(riego1, LOW);
     digitalWrite(riego2, LOW);
  }
}
```

Figura 46: *Código de manejo de bomba de riego*

Fuente*: Elaboración propia*

3.4.7. Control de sistema de alarma

Para el control de la alarma se usa una variable llamada “sensor”, almacena el valor del pin A5 del Arduino, este pin también está configurado como análogo.

```
Int sensor =analogRead(A5);
```

El rango de valores que maneja el pin de entrada va desde 0 a 100, mientras que la señal de salida de los sensores infrarrojos ubicados en la ventana del segundo piso va de 0 a 5 voltios, donde 0 voltios indica que el sensor está enviando y recibiendo la señal infrarroja y 5 voltios indica que el sensor fue activado, quiere decir que la señal infrarroja fue interrumpida, esto activará el pin de entrada del Arduino el cual activará una alarma.

Si el sensor está por debajo de su umbral, se activará la alarma poniendo el pin “pin_corte” en alto

```
If (sensor<25)

    digitalWrite(pin_corte,High)
```

Para desactivar la alarma, la variable “val” tiene que recibir la cadena de texto “alap”, el pin “pin_corte” se pone en estado bajo, apagando la sirena

3.4.8. Control de detección de gas

Según se muestra en la figura 47, para el control del sensor de gas se utiliza una variable del tipo bool, almacena el estado de la variable de entrada donde se encuentra conectado el sensor de gas al Arduino.

Si el sensor detecta gas, la variable mq_estado cambiara a “true” lo que ocasionará el encendido del led de color rojo, que será indicativo que hay fuga de gas.

```
//---------------SENSOR DE GAS ----------------
   boolean mq_estado = digitalRead(pin_mq);//Leemos el sensor
        if(mq_estado) //si la salida del sensor es 1
        {

          digitalWrite(focogas, HIGH);

        }
        else //si la salida del sensor es 0
        {
          digitalWrite(focogas, LOW);

        }
```

***Figura 47**: Código de control de sensor de gas*

***Fuente**: Elaboración propia*

3.4.9. Control de alta tensión

Para el control de los circuitos de alta tensión se procede de la siguiente forma, la variable “val” es validada en todos los casos, en caso tenga el valor de “foco1”, se leerá el estado del pin luzBano, si es 0 se encenderá la luz, caso contrario la luz se apagará.(Figura 48)

```
// -------------------------  ALTA TENSION--------------------
 //--- Control de encendido/apagado de luz de baño
    String val=Serial.readString();
    if (val == "foco1")  //variable recibida de aplicacion
    {

      int estado=digitalRead(luzBano);//leemos estado actual de pin luzBano
      if (estado==0)// Si esta apagado
        {
          digitalWrite(luzBano, HIGH); //activamos pin luzBano

        }
      else
        {
          digitalWrite(luzBano, LOW);//desactivamos pin luzBano
        }
    }
//---------------------------
```

***Figura 48**: Código control de luz de baño*

***Fuente**: Elaboración propia*

Lo mismo se valida para el encendido/apagado del ventilador (figura 49).

```
//--- Control de encendido/apagado de ventilador
if (val == "ve1") //variable recibida de aplicacion
    {

      int estado=digitalRead(ventilador);
      if (estado==0)
       {
          digitalWrite(ventilador, HIGH);

       }
      else
       {
          digitalWrite(ventilador, LOW);

       }

    }

  //---------------------------
```

Figura 49: *Código de control de ventilador*

Fuente*: Elaboración propia*

3.4.10. Circuitos de baja tensión

Para el control de los circuitos de baja tensión se utiliza el siguiente código:

a) Control de radiador de calefacción (sala)

Para que se encienda/apague el sistema de calefacción, la variable "val" debe ser "rd1", si coincide, se lee el estado actual del pin "radiadorSala", si está en estado alto (1), cambiará su estado a bajo (0) y viceversa

b) **Control de encendido/apagado de lámpara 1 y lámpara 2**

En este caso la variable val debe de ser "lam1", se realiza la misma validación que el calefactor validando en este caso el pin asociado a la variable lamp1 y lamp2 respectivamente.

c) **Control de encendido /apagado de foco 1 y foco 2 del patio trasero**

Para el control de las luces del patio trasero, la variable val debe de ser "fpat1" para la primera luz y "fpat2" para la segunda luz, se realizan las mismas validaciones que el caso anterior cambiando de estado a las variables focoPatio1 y focoPatio2

d) **Control de encendido /apagado de foco de entrada y foco de garaje (perímetro)** Para este caso, la variable val debe ser "fe" para el focoEntrada y "fg" para el foco del garaje.

e) Control que permite abrir / cerrar la puerta de garaje

Si "val" tiene el valor de "abgara", el motor girará para un sentido abriendo la puerta del garaje, si "val" tiene el valor de "cigara", el motor del garaje girará en sentido contrario cerrando la puerta del garaje. Esto se logra enviando los valores 1 y 0 por los pines ptaGarage1 y ptaGarage2 según la tabla 3

VARIABLES	VALOR	
ptaGarage1	0	1
PtaGarage2	1	0
	giro derecha	giro izquierda

Tabla 3: *Estados posibles de variables de control de giro de motor de puerta garaje*

Fuente*: Elaboración propia*

f) Control que permite abrir / cerrar la puerta de patio y ascensor.

Caso similar sucede con el control de la puerta del patio y el motor de subida /bajada del ascensor, si las variables ptaPatio1 vale 0 y ptaPatio2 vale 1, se abre la puerta del patio, si ptaPatio1 vale 1 y ptaPatio2 vale 0, la puerta se

cerrará, lo mismo sucede con el motor del ascensor; estos estados se pueden ver en las tablas 4 y 5

VARIABLES	VALOR	
ptaPatio1	0	1
ptaPatio2	1	0
	Abre puerta	Cierra puerta

Tabla 4: *Estados posibles de variables de control de giro de motor de puerta patio*

Fuente*: Elaboración propia*

VARIABLES	VALOR	
Ascen1	0	1
Ascen1	1	0
	Sube ascensor	Baja ascensor

Tabla 5: *Estados posibles de variables de control de giro de motor de ascensor*

Fuente*: Elaboración propia*

g) Control de servos de puerta de ascensor.

El código que permite el control de los servos se detalla en la figura 50, Si la variable "val" recibe la cadena de texto Ptasc, se leerá el estado del pin en el cual está ubicado el servo motor que controla la puerta del ascensor, si este valor es 0 , quiere decir que el pin de Arduino está a 0 grados, lo cual es validado y con el comando ServoAsc.write (90), se envía al Arduino el valor de 90, donde el servo motor se moverá 90 grados abriendo la puerta

Lógica similar se utiliza para el control de la puerta del baño:

```
//--------------CONTROL DE SERVOS----------------------

//CONTROL DE ABRIR/CERRAR PUERTA DE ASCENSOR
    if (val == "Ptasc")
    {

      int estado=digitalRead(puertaAsc);
      if (estado==0)
       {
          digitalWrite(puertaAsc, HIGH);//PIN QUE PERMITE VER ESTADO
          ServoAsc.write(90);//GIRA EJE A 90 GRADOS
       }
      else
       {
          digitalWrite(puertaAsc, LOW);
          ServoAsc.write(0);//GIRA EJE A POSICION INICIAL
       }

    }
```

***Figura 50**: Código de control de servos*
***Fuente**: Elaboración propia*

¿Cómo hace el sistema Web para saber cuándo un pin está activo a inactivo?

En Arduino, como en cualquier sistema digital, el estado activo se caracteriza por tener valores de 5 voltios, mientras que para el estado pasivo, el valor que corresponde es el de 0 voltios.

Esta característica es utilizada para verificar los estados de los pines de Arduino y enviarlos al sistema Web.

El procedimiento usado es almacenar en variables el estado actual de los pines de salida, estos valores serán 0 para pasivo y 1 para activo.

Luego se utiliza un buffer del tipo char el cual almacenará los valores de las variables a y b.

El comando sprintf permite convertir las variables del tipo int a y b a tipo caracter y los almacena en el arreglo “buffer”, luego de ello se itera tantas veces como valores en el arreglo imprimiendo por el monitor serie los valores del buffer, estos valores serán recogidos por la aplicación Web y serán validados los cuales serán utilizados para mostrar los iconos de activo o inactivo de cada dispositivo.

La sintaxis del comando sprintf se detalla en la figura 51.

```
sprintf(char* buffer, char* CadenaFormato, object Valor);
/*            |             |                    \_ Lista de valores que se van a conver
              |             \_ cadena con la máscara de los valores a convertir
              \_ Buffer donde devolveremos el resultado de los valores convertidos */
```

***Figura 51**: Sintaxis de sprintf*

***Fuente**: Elaboración propia*

En la figura 52 se muestra el código que valida los estados de los pines, el comando sprintf almacena en arreglo “buffer” los estados de los pines, el comando serial.println escribe en el puerto serie los valores actuales de los pines asociados a luzBano y a puertaBan

```
//Verifica estado de pin luzBano y puertaBan
if (val == "b1")
    {
      int a=digitalRead(luzBano);
      int b=digitalRead(puertaBan);

      char buffer [50];
      i=sprintf (buffer, "%d%d", a, b);
      for(int l= 0; l<=i; l++)
       {
          Serial.print(buffer[l]);
       }

    }
```

***Figura 52:** Verificación de estados de pines (dispositivos baño)*

***Fuente**: Elaboración propia*

La misma lógica es utilizada para la verificación de los estados de los pines de salida utilizados para encender/apagar los dispositivos de la sala y la lectura de estados de las variables que controlan los estados de los motores (motor de puerta de patio, subir/bajar ascensor y puerta de garaje)

Envío de señales infrarrojas para controlar la televisión.

Para el control de la televisión se ha implementado varias funciones que se encargan de enviar los códigos de frecuencia que permitirán activar las diferentes funciones de este aparato.

Si la variable "val" almacena el valor "on", se ejecutará la función "encApa", ejecutará el comando irsend.sendRaw(onoff,52,38) la cual encenderá la televisión.

VALOR DE "VAL"	FUNCION ASOCIADA	EJECUTA	ACCION
ON	encApa()	irsend.sendRaw(onoff,52,38);	enciende tv
1	ch1()	irsend.sendRaw(canal1,52,38);	canal 1
2	ch2()	irsend.sendRaw(canal2,52,38);	canal 2
3	ch3()	irsend.sendRaw(canal3,52,38);	canal 3
4	ch4()	irsend.sendRaw(canal4,52,38);	canal 4
5	ch5()	irsend.sendRaw(canal5,52,38);	canal 5
6	ch6()	irsend.sendRaw(canal6,52,38);	canal 6
7	ch7()	irsend.sendRaw(canal7,52,38);	canal 7
8	ch8()	irsend.sendRaw(canal8,52,38);	canal 8
9	ch9()	irsend.sendRaw(canal9,52,38);	canal 9

0	ch0()	irsend.sendRaw(canal0,52,38);	canal 0
sv	svo()	irsend.sendRaw(subeVol,52,38);	sube volumen
bv	bvo()	irsend.sendRaw(bajaVol,52,38);	baja volumen
sc	sch()	irsend.sendRaw(subeCh,52,38);	sube canal
bc	bch()	irsend.sendRaw(bajaCh,52,38);	baja canal

Tabla 6: *Resumen de funciones y comandos utilizados para controlar la televisión*

Fuente*: Elaboración propia*

Con el comando irsend enviamos 3 parámetros:

Parámetro onoff: envía la variable del tipo int "onoff" (figura 53), aquí se incluye códigos RAW que representan la frecuencia que corresponde al botón de encendido del control remoto de la tv.

```
unsigned int onoff[52] = {3800, 3950, 500, 1950, 500, 1950, 500, 1950,
500, 1950, 450,  1050, 450,  1000, 500,  1950, 500,  1000, 450, 2000,
450,  1000, 500,  1950, 500,  1000, 500,  1000, 450,  1000, 550,  950,
500,  1000, 450,  2000, 500,  1950, 450, 1000, 500,  1950, 500,  1000,
500,  1950, 500,  950, 500,  1950, 500};
```

Figura 53: *Variable onoff con su código RAW*

Fuente*: Elaboración propia*

Parámetro 52: es la longitud del arreglo onoff.

Parámetro 38: simboliza la frecuencia del diodo emisor, normalmente 38kHz

3.5. Aplicación Web

La arquitectura de la aplicación Web se basa en el patrón MVC (Modelo – Vista – Controlador) (Microsoft, 2016)

El MVC o Modelo-Vista-Controlador es un patrón de arquitectura de software que, utilizando 3 componentes (Vistas, Modelos y Controladores) separa la lógica de la aplicación, de la lógica de la vista en una aplicación. Es una arquitectura importante puesto que se utiliza tanto en componentes gráficos básicos hasta sistemas empresariales.

La razón por la que se usa este patrón en el proyecto, es que nos permite separar los componentes de la aplicación dependiendo de la responsabilidad que tienen, esto significa que cuando se hace un cambio en alguna parte del código, esto no afecte otra parte del mismo. Por ejemplo, si modifico la Base de Datos, sólo deberíamos modificar el modelo que es quién se encarga de los datos y el resto de la aplicación debería permanecer intacta.

3.5.1. El modelo.

El modelo de nuestra aplicación está representado por la base de datos utilizada para almacenar a los usuarios del sistema y sus contraseñas, como se indicó en un inicio, solo se tiene implementada una tabla llamada “Usuarios”.

3.5.2. Los controladores.

Los controladores del presente proyecto se detallan en la figura 54.

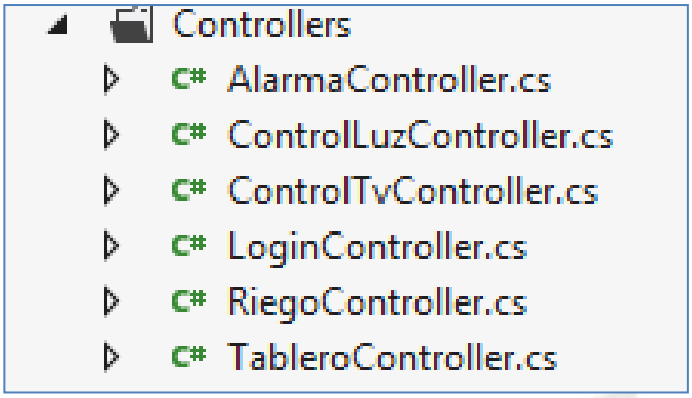

Figura 54: *Controladores del proyecto*

Fuente*: Elaboración propia*

a) Controlador AlarmaController.

Este controlador se encarga de apagar la alarma en caso se active por alguna intrusión.

En la figura 55 se detalla el código utilizado, se explica a continuación.

Con el comando ardo.portname=”COM3”, se define que se utilizará el com 3 para la comunicación de la placa Arduino con la pc.

Con el comando ardo.BaudRate se configura el valor de la tasa de transferencia de bits por segundo por el puerto serie COM3, en este caso un valor estándar es 9600.

La variable “blue” almacena la cadena “alap”

El comando “Open” abre la conexión serial.

El comando ardo.Write envía la cadena “alap” a la placa Arduino, luego se cierra la conexión con ardo.Close()

```
public void Alarma1()
{
    ardo.PortName = "COM3";
    ardo.BaudRate = 9600;

    string blue = "alap";
    ardo.Open();
    ardo.Write(blue);

    //System.Threading.Thread.Sleep(1000);
    ardo.Close();


}
```

***Figura 55**: Método Alarma1*

***Fuente**: Elaboración propia*

b) Controlador LuzController.

Este controlador se encarga del control de las luces de bajo voltaje.

En la figura 56 se muestra el código utilizado para el control de la lámpara de la sala; se envía la cadena de texto "lam1" con el comando **ardo.write(lam)** al Arduino, el cual validará y encenderá la luz de la lámpara 1 de la sala

```
//BLOQUE QUE PERMITE EL ENCENDIDO DE LAS LAMPARAS DE LA SALA

        public void Lampara1_Sala()
        {
            ardo.PortName = "COM3";
            ardo.BaudRate = 9600;

            string lam = "lam1";
            ardo.Open();
            ardo.Write(lam);

            //System.Threading.Thread.Sleep(1000);
            ardo.Close();

        }
```

***Figura 56**: Método Lampara1_Sala*

***Fuente**: Elaboración propia*

Lógica similar se aplica para las luces de la lámpara 2 de la sala, luces 1 y 2 del patio y las 2 luces del perímetro.

A continuación en la tabla 7, un resumen de los valores enviados al Arduino según el dispositivo que se quiere controlar.

FUNCION	ACCION	STRING ENVIADO
Lampara1_Sala()	enciende lámpara 1 sala	lam1
Lampara2_Sala()	enciende lámpara 2 sala	lam2
Luz_Patio1()	enciende luz 1 patio	fpat1
Luz_Patio2()	enciende luz 2 patio	fpat2
PeriLuzGara()	enciende luz perímetro garaje	fg
PeriLuzPrinc()	enciende luz puerta principal	fe

***Tabla 7**: String enviados a Arduino*

***Fuente**: Elaboración propia*

En este mismo controlador se encuentran también las funciones que permiten la sincronización de los valores actuales de estado de los diferentes pines de salida del Arduino.

En la figura 57 se muestra el código de sincronización, éste configura el puerto y los bits por segundo, se envía la cadena “ls1” al Arduino luego con la instrucción **Ardo.readexisting ()** se verifica si hay datos en el puerto serie del Arduino, de existir datos son trabajados en la vista para la posterior actualización de los iconos de estado de los diferentes dispositivos, esto se logra con el valor retornado en la variable “POT”

```
//BLOQUE QUE PERMITE SINCRONIZAR EL ESTADO DE LAS LAMPARAS DE LA SALA

        public string Sincro_Luces()
        {
            ardo.PortName = "COM3";
            ardo.BaudRate = 9600;

            string sincro = "ls1";
            ardo.Open();
            ardo.Write(sincro);
            System.Threading.Thread.Sleep(2000);
            string POT = ardo.ReadExisting();
            ardo.Close();
            return POT;

        }
```

Figura 57: *Método Sincronización de luces*

Fuente*: Elaboración propia*

La misma lógica es usada para la sincronización de las luces del patio (función Sincro_patio()) y las luces del perímetro (función Sincro_Perimetro()).

c) Controlador ControlTvController,

En este controlador se implementan las diferentes funciones que permiten enviar al televisor los comandos que permiten por ejemplo cambiar de canal o subir/bajar volumen

En la figura 58 se muestra el código utilizado, se configura igualmente el puerto COM3 y su transferencia de datos, luego se envía al Arduino el valor de cadena “on” el cual será procesado en el código del Arduino dando lugar al encendido de la tv.

```
public void Enc_Tv()
{
    ardo.PortName = "COM3";
    ardo.BaudRate = 9600;

    string sincro = "on";
    ardo.Open();
    ardo.Write(sincro);
    ardo.Close();


}
```

Figura 58: *Método para encender TV*

Fuente*: Elaboración propia*

En la tabla 8 se detalla la relación de funciones usadas en este controlador, cada una envía al Arduino un parámetro distinto que permitirá el control de la tv.

FUNCION	PARAMETRO ENVIADO
Enc_Tv()	on
Ca1()	1
Ca2()	2
Ca3()	3
Ca4()	4
Ca5()	5
Ca6()	6
Ca7()	7
Ca8()	8
Ca9()	9
Ca0()	0
subVol()	sv
bajaVol()	bv
subeCanal()	sc
bajaCanal()	bc

Tabla 8: *Funciones de ControlTvController*

Fuente*: Elaboración propia*

d) Controlador LoginController.

Este controlador se encarga del manejo del login del usuario a nivel de base de datos, básicamente cuenta con 2 métodos: Login y Salir.

En la figura 59 se detalla el código utilizado para estos 2 métodos.

El método Login trabaja con la vista Login.cshtml, la cual le envía los parámetros del usuario y password almacenándolos en las variables "usuarionombre" y "password".

Estas variables son comparadas por base de datos verificando que exista el nombre de usuario y password.

El método Salir() al activarse sale del sistema y redirecciona a la vista "login".

```
[HttpPost]
public ActionResult Login(FormCollection form)
{
    string usuarionombre = form["usuario"].ToString();
    string password = form["password"].ToString();

    var usr = (from u in bd.Usuarios
               where u.Nombre == usuarionombre && u.Password == password && u.EsActivo == true
               select u).FirstOrDefault();
    if (usr != null)
    {
        FormsAuthentication.SetAuthCookie(usr.Nombre, false);
        return RedirectToAction("index", "Tablero");
    }
    TempData["Message"] = "Usuario o password incorrectos";
    return View();
}


public ActionResult Salir()
{
    FormsAuthentication.SignOut();
    return RedirectToAction("Login");
}
```

***Figura 59**: Controlador Login*

***Fuente**: Elaboración propia*

e) Controlador RiegoController.

Este controlador almacena 2 métodos: Riego1 ( ) y Riego2 ( ).

El primero se encarga de encender el sistema de riego manualmente enviando al Arduino la cadena “preri” la cual será validada y encenderá la bomba de riego.

El segundo método (figura 60), se encarga de apagar el sistema de riego, enviando al puerto serie la cadena “apa” la cual será validada por el Arduino apagando la bomba de riego.

```
public void Riego1()
{
    ardo.PortName = "COM3";//elegimos el COM3
    ardo.BaudRate = 9600;  //Tasa de transferencia de bits por segundo
    string blue = "preri"; //cadena que sera validada en el codigo Arduino
                           //(en este caso prende el sistema de riego)
    ardo.Open();           //Abrimos conexion con el Arduino
    ardo.Write(blue);      //enviamos el valor de la variable al puerto serie
    ardo.Close();          //cerramos la conexion


}

public void Riego2()
{
    ardo.PortName = "COM3";//elegimos el COM3
    ardo.BaudRate = 9600;  //Tasa de transferencia de bits por segundo
    string blue = "apa";   //cadena que sera validada en el codigo Arduino
                           //(en este caso prende el sistema de riego)
    ardo.Open();           //Abrimos conexion con el Arduino
    ardo.Write(blue);      //enviamos el valor de la variable al puerto serie
    ardo.Close();          //cerramos la conexion


}
```

Figura 60: *Métodos Riego1 y Riego2*

Fuente*: Elaboración propia*

f) Controlador TableroController.

Este controlador declara 11 métodos. Adicionalmente se declaran 4 métodos más que permiten sincronizar y actualizar los estados de cada pin del Arduino, los métodos son:

FUNCION	PARAMETRO ENVIADO	ACCION
Foco_Baño1()	"foco1"	enciende foco de baño
Ventilador()	"ve1"	enciende ventilador
Radiador_Sala()	"rd1"	enciende calefactor
Garage1()	"abgara"	abre puerta de garage
Garage2()	"cigara"	cierra puerta de garage
Puerta_cocina1()	"abpat"	abre puerta de cocina
Puerta_cocina2()	"cipat"	cierra puerta de cocina
Ele_Sube()	"subasc"	sube elevador
Ele_Baja()	"bajasc"	baja elevador
Puerta_Asc()	"Ptasc"	abre/cierra puerta ascensor
Puerta_Baño1()	"Ptaba"	abre/cierra puerta baño
Sincro_estado()	"b1"	sincroniza los datos de Arduino
Sincro_sala()	"r1"	sincroniza el estado de dispositivos de sala
Sincro_Garage()	"r2"	sincroniza el estado del garage(abierto/cerrado)
Sincro_Cocina()	"r3"	sincroniza el estado de la puerta de patio(abierta/cerrada)

Tabla 9: *Métodos utilizados en Controlador TableroController*

Fuente*: Elaboración propia*

3.5.3. Vistas.

Las vistas son la parte de la aplicación donde generamos el código HTML; estos códigos son procesados por el navegador el cual nos mostrará las diferentes páginas Web del proyecto.

El proyecto define 15 vistas, en la figura 61 se enumeran las vistas utilizadas.

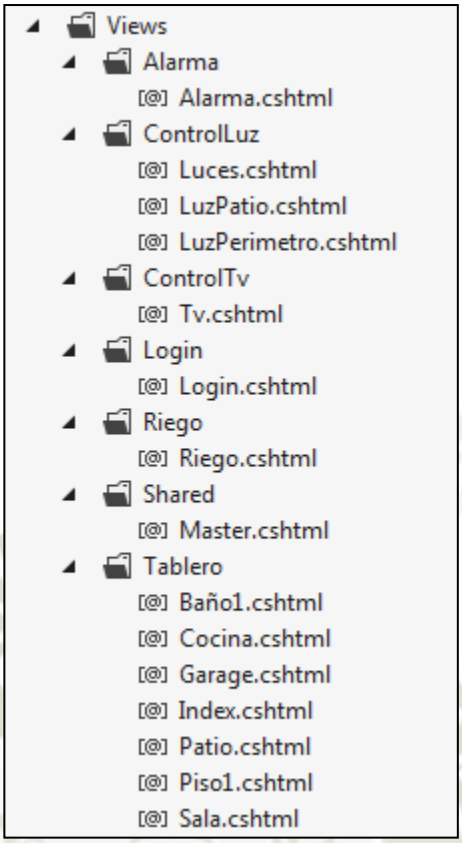

Figura 61: *Listado de vistas implementadas*

Fuente*: Elaboración propia*

a) Vista Master.cshtml

Esta es la página maestra de la aplicación, es la que se usa en cada una de las vistas; aquí se define el menú del tipo "Side Bar", los links "nav-bar" y el control que autentica al usuario.

Para aplicar el estilo se utiliza la librería de bootstrap.

El comando @RenderBody permite renderizar o cargar la vista secundaria.

b) Vista Alarma.cshtml

Esta vista contiene el código html y código javascript que permite apagar la alarma en caso haya sido activada.

El evento onclick define el método “Apaga_Alarma( )” que permitirá apagar la sirena al hacer click sobre la imagen “ventana.jpg” (figura 62).

```
<div class="row-fluid col-md-offset-1 col-lg-offset-2">
    <div class="span12">
        <div id="counter">
            <input type="hidden" name="action" value="0" id="accion1" />
             <img src="../../Diseño/img/../../Diseño/img/ventana.jpg" id="img1" alt="bg" onclick="Apaga_Alarma();"/>
        </div>
    </div>
</div>
```

***Figura 62**: Evento Onclick “Apaga Alarma”*

***Fuente**: Elaboración propia*

El script se detalla en la figura 63; para apagar la alarma llama al método Alarma1 del controlador AlarmaController.

El comando getElementById(“accion1”).value==0 compara si el atributo value del div counter es 0 , de ser afirmativo, ejecuta el método “Alarma1” del controlador AlarmaController

```
<script>

    function Apaga_Alarma() {
        if (document.getElementById("accion1").value == 0) {
            document.getElementById("img1").title = "APAGA ALARMA";
            $.get("@Url.Action("Alarma1","Alarma")",
            {

            });
        }
    }
</script>
```

***Figura 63:** Script Apaga Alarma*

***Fuente**: Elaboración propia*

c) Vista Luces.cshtml.

Esta vista contiene el código html y código javascript que permite visualizar las 2 lámparas de la sala, encenderlas y apagarlas.

Los eventos onclick definen los métodos "Enciende_lampara1( )" y "Enciende_lampara2( )" que permitirá apagar y encender las 2 lámparas (figura 64).

```
<div id="counter">
    <input type="hidden" name="action" value="0" id="accion1" />
    <input type="hidden" name="action" value="0" id="accion2" />
    <img class="overLamp1" src="../../Diseño/img/lampara_off.png" id="img1"   title="ENCIENDE LAMPARA1" onclick="Enciende_lampara1();" />
    <img class="overLamp2" src="../../Diseño/img/lampara_off.png" id="img2"   title="ENCIENDE LAMPARA2" onclick="Enciende_lampara2();"/>
    <img src="../../Diseño/img/../../Diseño/img/sala.png" alt="bg" />
</div>
```

***Figura 64**: Eventos onclick lámparas*

***Fuente**: Elaboración propia*

El script detallado en la figura 65 se encarga de ejecutar el método Sincro_Luces del controlador ControlLuz el cual almacenará en las variables letra1 y letra2 los valores enviados desde el controlador los cuales fueron traídos del Arduino que representan los valores actuales de los estados de los pines del Arduino, es así como se determina en qué estado se encuentran estos pines actualizando el icono que representa a lámpara encendida / apagada.

```
$(document).ready(function (e) {
    $.get("@Url.Action("Sincro_Luces","ControlLuz")", function (data) {
        var letra1 = data.charAt(0);
        var letra2 = data.charAt(1);
          if (letra1 == '1') {
            image = "../../Diseño/img/lampara_on.png";
            document.getElementById("img1").src = image;
            document.getElementById("accion1").value = 1

           }

        else {
            image = "../../Diseño/img/lampara_off.png";
            document.getElementById("img1").src = image;
           }
        if (letra2 == '1') {
            image = "../../Diseño/img/lampara_on.png";
            document.getElementById("img2").src = image;
            document.getElementById("accion2").value = 1
        }

        else {
            image = "../../Diseño/img/lampara_off.png";
            document.getElementById("img2").src = image;
        }
    });
});
```

Figura 65: *Script sincronización de lámparas al cargar pagina*

Fuente*: Elaboración propia*

En la tabla 10 se muestra un resumen de las funciones implementadas que tienen una codificación similar a la descrita anteriormente.

VISTA	FUNCION SINCRONIZA	FUNCION EJECUTA	FUNCION DE CONTROLADOR	CONTROLADOR	ACCION
LuzPatio.cshtml	SincroPatio	EnciendeFocoPatio1()	LuzPatio1	ControlLuz	Enciende foco 1 de patio
		EnciendeFocoPatio2()	LuzPatio2	ControlLuz	Enciende foco 2 de patio
LuzPerimetro.cshtml	SincroPerimetro	LuzPeriGarage()	PeriLuzPrinc	ControlLuz	Enciende luz perimetro garage
		LuzPeriPrinc()	PeriLuzGara	ControlLuz	Enciende luz perimetro puerta principal
Riego.cshtml	no corresponde	Enciende_Riego()	Riego1	Riego	Enciende/apaga bomba de riego
Baño1.cshtml	Sincro_Estado	ActializaFoco()	Foco_Baño1	Tablero	Enciende/apaga luz de baño
	Sincro_Estado	Actualiza_Puerta()	Puerta_Baño1	Tablero	Abre/cierra puerta de baño
Cocina.cshtml	Sincro_Cocina	Abre_Puerta_cocina()	Puerta_cocina1	Tablero	abre puerta corrediza de patio
			Puerta_cocina2	Tablero	cierra puerta corrediza de patio
garage.cshtml	Sincro_Garage	Ejecuta_Garage()	Garage1()	Tablero	abre puerta garage
			Garage2()	Tablero	cierra puerta garage
Patio.cshtml	Sincro_Luces	Enciende_Lampara1()	lampara1_sala	ControlLuz	Enciende/apaga luz lampara1 sala
		Enciende_Lampara2()	lampara2_sala	ControlLuz	Enciende/apaga luz lampara2 sala
Sala.cshtml	Sincro_Sala	EnciendeCalefaccion()	RadiadorSala()	Tablero	Enciende/apaga calefactor
		Sinc_Puerta_Asc()	Puerta_Asc()	Tablero	Abre/cierra puerta ascensor
		Sinc_Elevador_Up()	Ele_Sube()	Tablero	Sube elevador
			Ele_Baja()	Tablero	Baja elevador
		Sinc_ventila()	Ventilador()	Tablero	Enciende/apaga ventilador

***Tabla 10**: Resumen de vistas con codificación similar*

***Fuente**: Elaboración propia*

d) Vista Tv.cshtml.

En esta vista se implementa la visualización del control remoto del televisor el cual permitirá manipular este equipo a través de la aplicación.

Cada tecla ejecuta una función definida en la sección de scripts de la vista.

Para generar el área de acción mediante click de mouse se está utilizando mapas con coordenadas (figura 66)

Cada mapa define un evento onclick el que a su vez llama a diferentes métodos que generaran desde el encendido de la televisión hasta subir o bajar volumen.

```
<map name="Tv">


    @*power 1,2,3*@
    <area shape="poly" coords="28,16,61,15,63,23,59,35,59,42,57,51,44,55,32,51,29,40,28,16" title="ON" onclick="Enciende_Tv();" >
    <area shape="poly" coords="33,86,53,86,59,90,60,94,57,103,34,102,30,100,28,95,33,86" title="UNO" onclick="Canal_Uno();" >
    <area shape="poly" coords="83,87,106,88,113,94,108,103,86,102,86,102,82,101,80,95,83,87"title="DOS" onclick="Canal_Dos();">
    <area shape="poly" coords="133,87,156,87,163,96,159,102,136,102,130,95,132,89,133,87" title="TRES" onclick="Canal_Tres();">
   @* 4,5,6*@
    <area shape="poly" coords="30,134,52,134,61,142,59,149,32,149,27,145,28,141,30,134" title="CUATRO" onclick="Canal_Cuatro();" >
    <area shape="poly" coords="83,135,104,134,111,143,108,150,84,150,79,145,79,140,83,135" title="CINCO" onclick="Canal_Cinco();">
    <area shape="poly" coords="132,135,155,135,163,143,158,150,135,150,130,145,128,139,132,135" title="SEIS" onclick="Canal_Seis();">
    @*7,8,9*@
    <area shape="poly" coords="31,187,54,187,61,192,60,198,58,203,35,203,30,198,29,192,31,187" title="SIETE" onclick="Canal_Siete();" >
    <area shape="poly" coords="83,187,108,187,112,195,109,203,85,203,79,198,79,192,83,187" title="OCHO" onclick="Canal_Ocho();">
    <area shape="poly" coords="132,187,157,187,163,195,158,203,135,203,130,198,128,193,132,187" title="NUEVE" onclick="Canal_Nueve();">
    @*0*@
    <area shape="poly" coords="82,238,107,238,113,246,108,254,86,254,80,249,80,249,82,238" title="CERO" onclick="Canal_Cero();" >
    @*canal *@
    <area shape="poly" coords="69,299,82,294,95,291,109,294,122,301,113,311,104,306,88,307,81,310,69,299" title="Canal +" onclick="Sube_Canal();">
    <area shape="poly" coords="78,360,91,365,101,365,111,360,122,370,106,378,84,377,70,370,78,360" title="Canal -" onclick="Baja_Canal();">
    @*volumen *@
    <area shape="poly" coords="60,307,72,320,66,328,67,342,72,351,63,363,53,345,52,325,60,307" title="Volumen -"  onclick="Baja_Volumen();">
    <area shape="poly" coords="128,309,137,324,139,343,138,350,131,363,118,353,123,341,123,330,128,309" title="Volumen +"  onclick="Sube_Volumen();">

</map>
```

***Figura 66**: Mapas en base a polígonos*

***Fuente**: Elaboración propia*

En la figura 67 se muestran algunos de los script que ejecutan las diferentes órdenes, por ejemplo tenemos la función “Enciende_Tv( )”la cual llama a la función “enc_Tv” del controlador ControlTvController

La misma lógica se usa para los canales y el control del volumen.

```
<script>


    function Enciende_Tv() {

        $.get("@Url.Action("Enc_Tv","ControlTv")",
     {

     });
    }
    function Canal_Uno() {

        $.get("@Url.Action("Ca1","ControlTv")",
     {

     });
    }


    function Canal_Dos() {

        $.get("@Url.Action("Ca2","ControlTv")",
     {

     });
    }
```

Figura 67: *Scripts para el control de TV*

Fuente*: Elaboración propia*

e) Vista Login.cshtml.

En esta vista se implementa el código html que permite visualizar la página inicial de login.

Al momento que se ejecuta la aplicación, esta es la primera pantalla que aparece.

Son dos cuadros de texto los cuales solicitan el nombre de usuario y password, en caso de que se ingrese un password errado, el código html valida esto, y muestra el mensaje "usuario o password incorrectos", esto se valida en el controlador LoginController".

Esta vista también utiliza como gestor de estilos a la librería de bootstrap que permite que sea responsiva (Figura 68).

```
<div class="row">
      <div class="col-sm-6 col-sm-offset-3 col-xs-8 col-xs-offset-2  myform-cont" >
          <div class="myform-top">
              <div class="myform-top-left ">
                <h3>Ingrese usuario y password</h3>
                   @if(TempData["Message"]!=null)
                        {
                            <div class="mydescription">
                                @TempData["Message"].ToString()
                            </div>
                        }
              </div>

          </div>
          <div class="myform-bottom">
            <form role="form" action="@Url.Content("/Login/Login/")" method="post" class="">
              <div class="form-group">
                  <input type="text" name="usuario" placeholder="Usuario..." class="form-control" id="form-username">
              </div>
              <div class="form-group">
                  <input type="password" name="password" placeholder="Contraseña..." class="form-control" id="form-password">
              </div>
              <button type="submit" class="mybtn">Entrar</button>
            </form>
          </div>
      </div>
</div>
```

Figura 68: *Código html de vista Login.cshtml*

Fuente*: Elaboración propia*

f) Vista Index.cshtml.

Esta vista contiene el código html que permite visualizar los links del primer y segundo piso, esta página es a la que redireccionará el sistema al momento que nos logueamos (figura 69)

Código Html:

```
<div >
     <div class="row ">
            <div class="col-lg-offset-1 col-lg-4 col-md-offset-1 col-md-4 col-sm-offset-1 col-sm-4"  >
                <a href="Piso1"><img src="../../Diseño/img/PISO1.png"    class="img-responsive" alt="" /></a>
            </div>
            <div class="col-lg-offset-1 col-lg-4 col-md-offset-1 col-md-4 col-sm-offset-1 col-sm-4"  >
                <a href="#"><img src="../../Diseño/img/PISO2.png"   class="img-responsive" alt="" /></a>
            </div>
      </div>

</div>
```

***Figura 69**: Vista index.cshtml*

***Fuente**: Elaboración propia*

g) Vista Piso1.cshtml.

Esta vista contiene el código html y código javascript que permite visualizar el mapa de todo el primer piso, cada ambiente del mapa está delimitado por mapa de coordenadas los cuales permiten que se pueda hacer click en las zonas delimitadas por éstas (figura 70).

El mapa tiene como nombre “PrimerPiso”

El tipo de mapa es “poly” de polígono, y sus coordenadas delimitadoras son los enteros declarados en la variable “coords”

```
<div class="col-lg-offset-4 col-lg-4 col-md-offset-4 col-md-4 col-sm-offset-4 col-sm-4" " >

    <img src="../../diseño/img/PRIMER_PISO.PNG" width="540" height="523" usemap="#PrimerPiso" class="img-responsive"  alt="">
    <map name="PrimerPiso">
        <area shape="poly" coords="49,150,142,156,137,261,35,256,49.150" href="Baño1" title="BAÑO" alt="BAÑO" >
        <area shape="poly" coords="145,154,506,170,511,270,364,260,362,308,135,296,145,154" href="Cocina" title="COCINA" alt="COCINA" >
        <area shape="poly" coords="364,264,509,269,516,409,364,399,364,264" href="Garage" title="GARAGE" alt="GARAGE">
        <area shape="poly" coords="36,256,138,265,132,299,359,308,361,400,286,505,8,495,36,256" href="Sala" title="SALA" alt="SALA">
        <area shape="poly" coords="64,21,502,41,507,169,48,148,64,21" href="Patio" title="PATIO" alt="PATIO">
    </map>
</div>
```

Figura 70: *Mapas en base a coordenadas*

Fuente*: Elaboración propia*

En la figura 71 se muestra la vista del 1er piso, el plano muestra cada ambiente del primer piso, los eventos click de cada ambiente se ejecutan gracias a la implementación de mapas a través de coordenadas.

Figura 71*: Mapa de primer piso*

Fuente*: Elaboración propia*

Para que la aplicación sea responsiva y el mapa responda de igual forma tanto en una tablet como en una pc, se utiliza la librería de javascript rwdImageMaps.js, esta librería permite mantener los límites de las coordenadas ya sea cuando se visualiza en una tablet o en una pc.

En la figura 72 se detalla el script donde se utiliza la función rwdImageMaps() que es parte de la librería rwdImageMaps.js

```
<script src="~/Diseño/js/jquery.rwdImageMaps.min.js"></script>

<script>
    $(document).ready(function (e) {
        $('img[usemap]').rwdImageMaps();

        $('area').on('click', function () {

        });
    });
</script>
```

***Figura 72**: Script mapas responsivos*

***Fuente**: Elaboración propia*

3.6. SUMARIO

Análisis de objetivos específicos:

OBJETIVO ESPECIFICO	ACCION	RESULTADO
Construir e implementar una estructura electrónica	Se construye las diferentes tarjetas electrónicas que permiten el control de todos los dispositivos.	Objetivo conseguido
Desarrollar un sistema web adaptable (sistema responsivo).	Se utiliza las librerías de bootstrap permitiendo así que el sistema Web pueda ser visualizado desde cualquier dispositivo que tenga instalado un navegador Web.	Objetivo conseguido
Controlar y automatizar los diferentes dispositivos domóticos de uso cotidiano.	Gracias al software y hardware construido se logró controlar los diferentes dispositivos como focos, ventiladores, etc.	Objetivo conseguido

***Tabla 11**: Análisis de objetivos específicos*

***Fuente**: Elaboración propia*

Análisis del objetivo general:

OBJETIVO GENERAL	ACCION	RESULTADO
Proponer una herramienta computacional orientada a dar ayuda a personas en silla de ruedas	En base a los objetivos específicos se construyó un sistema y/o prototipo que permita el objetivo general.	Objetivo conseguido

Tabla 12: *Análisis de objetivo general*

Fuente*: Elaboración propia*

CAPÍTULO 4: PRUEBAS Y RESULTADOS

4.1. Pruebas no funcionales

Una prueba no funcional es una prueba cuyo objetivo es la verificación de un requisito que especifica criterios que pueden usarse para juzgar la operación de un sistema (requisitos no funcionales) A continuación se detalla las pruebas realizadas al software y hardware:

4.1.1. Pruebas de usabilidad

Las pruebas de usabilidad se enfocan en validar que tan fácil de usar es la aplicación Las características evaluadas en la usabilidad incluyen:

- Facilidad de aprendizaje: Que tan fácil es para los usuarios realizar funciones básicas la primera vez que utilizan la aplicación.
- Eficiencia: Que tan rápido los usuarios experimentados pueden realizar sus tareas.
- Errores: Cuantos errores atribuibles al diseño comete el usuario, que tan severos son y que tan fácil es recuperarse de los mismos.
- Satisfacción: Que tanto le gusta (o desagrada) al usuario utilizar el sistema.

Para realizar las pruebas de usabilidad se realizó una encuesta, esta se aplicó a 10 personas discapacitadas que tenían movilidad normal en los miembros superiores, espalda y tórax y que usaban silla de ruedas.

Previamente se les dio una inducción a las diferentes opciones y la forma como se ingresa al sistema.

La encuesta usada se detalla en la figura 73.

ENCUESTA USABILIDAD APLICACIÓN DOMOTICA.

1.- ¿Fue fácil el ingreso a la aplicación?

SI ☐ NO ☐

2.- ¿En general, que opina de la velocidad de acceso y procesamiento de la aplicación?

RÁPIDO ☐ REGULAR ☐ LENTO ☐

3.- ¿Se presentaron errores al momento de usar el sistema?

SI ☐ NO ☐

4.- ¿En general que nivel de satisfacción usted tiene del sistema?

TOTALMENTE SATISFECHO ☐ SATISFECHO ☐ NO SATISFECHO ☐

Figura 73: *Encuesta para determinar la usabilidad del sistema*

Fuente. *Elaboración propia*

Resultados de la encuesta:

Facilidad de ingreso al sistema Web: Según se grafica en la figura 74, se observa que 8 personas de 10 encuestadas encuentran fácil el ingreso a la aplicación web, mientras que 2 personas no encuentran fácil el ingreso.

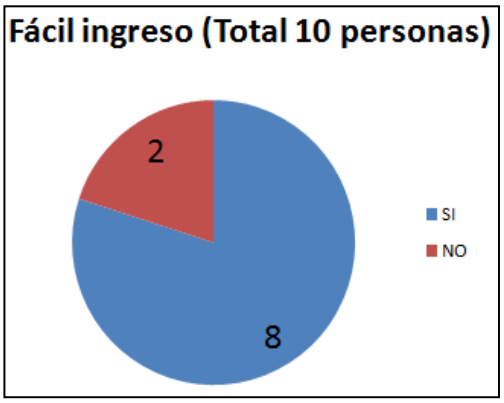

Figura 74: *Facilidad de ingreso al sistema*

Fuente. *Elaboración propia*

Velocidad de la aplicación: En la figura 75 se puede observar que 6 de 10 personas percibieron que el ingreso y la navegación entre las diferentes páginas de la aplicación Web es rápida, 3 percibieron velocidad regular y 1 percibió que es lenta.

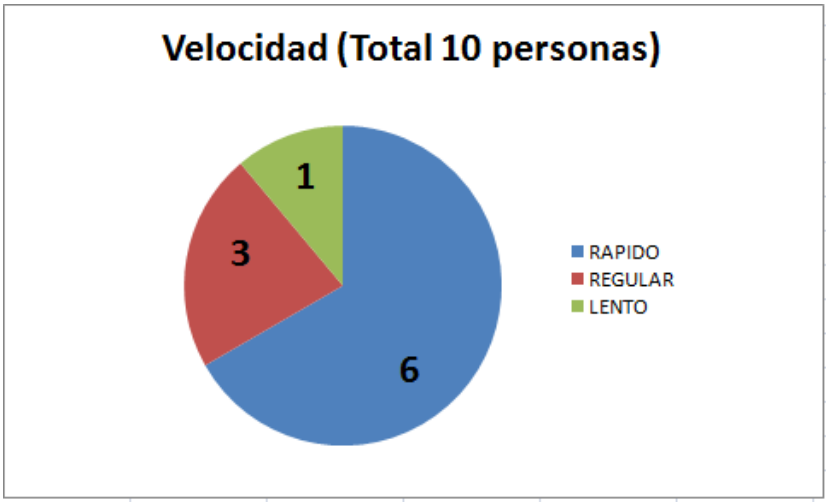

Figura 75: *Velocidad el sistema*

Fuente. *Elaboración propia*

Errores de la aplicación Web: En la figura 76 se observa que al momento de usar la aplicación Web 8 de cada 10 personas indican que no se presentaron errores al momento de navegar entre las diferentes páginas de la aplicación Web mientras que 2 personas si indican que tuvieron errores en la aplicación. Una vez detectados estos errores fueron subsanados.

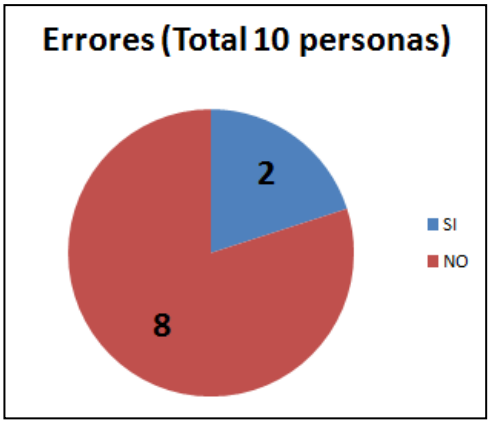

Figura 76: *Errores del sistema*

Fuente: *Elaboración propia*

Nivel de satisfacción: Como se observa en la figura 76, 6 de cada 10 personas indican que están totalmente satisfechas con el control domótico propuesto, mientras que 2 personas indican encontrarse satisfechas, no se tiene personas no satisfechas.

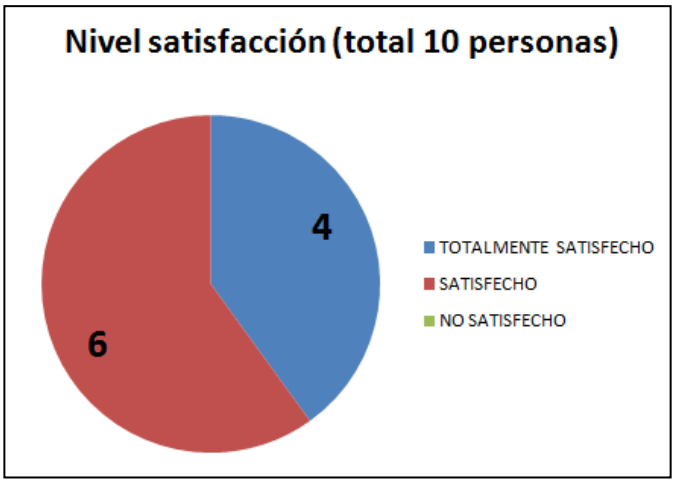

Figura 77: *Nivel de satisfacción en el uso y funcionamiento del sistema*

Fuente: *Elaboración propia*

4.1.2. Pruebas de recuperación

Las pruebas de recuperación se realizan para verificar que tan rápido y que tan bien se recupera una aplicación luego de experimentar un falló de hardware o software.

Estos fallos pueden ser:

- Interrupción de electricidad en el cliente.
- Interrupción de electricidad en el servidor: simular o iniciar procedimientos de pérdida de energía para el servidor.
- Interrupción de la comunicación en la red

a) Prueba de interrupción de electricidad en el cliente

Los clientes pensados para la conexión al sistema son tablets o celulares Smartphone, se realizó la prueba con una Tablet marca Acer.

En un primer momento el equipo está conectado al sistema, se esperó hasta que se acabe su batería, en un segundo momento se procedió a cargar y encender el equipo, se procedió a realizar la conexión al sistema y posterior autenticación a la base de datos, los resultados se detallan en la figura 78, el tiempo de reconexión a la red fue de 5 segundos mientras que el tiempo de reconexión al sistema fue de 4 segundos.

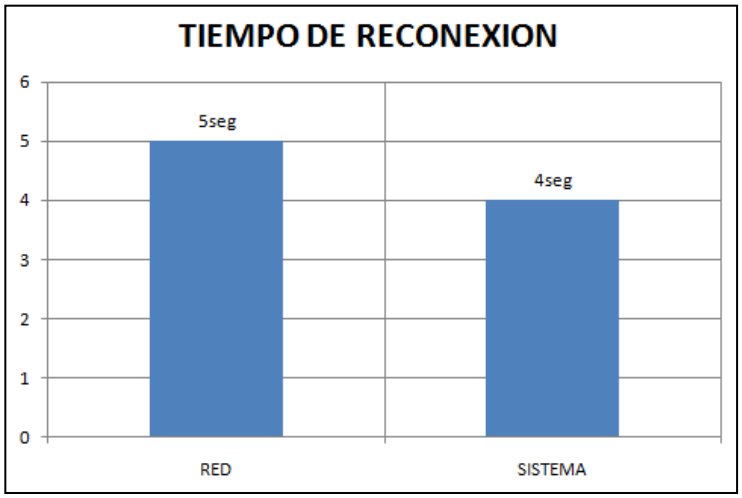

Figura 78: *Tiempo de reconexión del cliente*

Fuente*: Elaboración propia*

b) Interrupción de electricidad en el servidor

Para esta prueba se interrumpió el flujo de electricidad en el servidor Web, los resultados obtenidos, según se observa en la figura 79, el tiempo de demora en restablecer el sistema operativo en el servidor fue de 82 segundos, y el tiempo de disponibilidad del sistema 143 segundos.

Estos tiempos dependerán también de la velocidad y potencia del hardware del servidor.

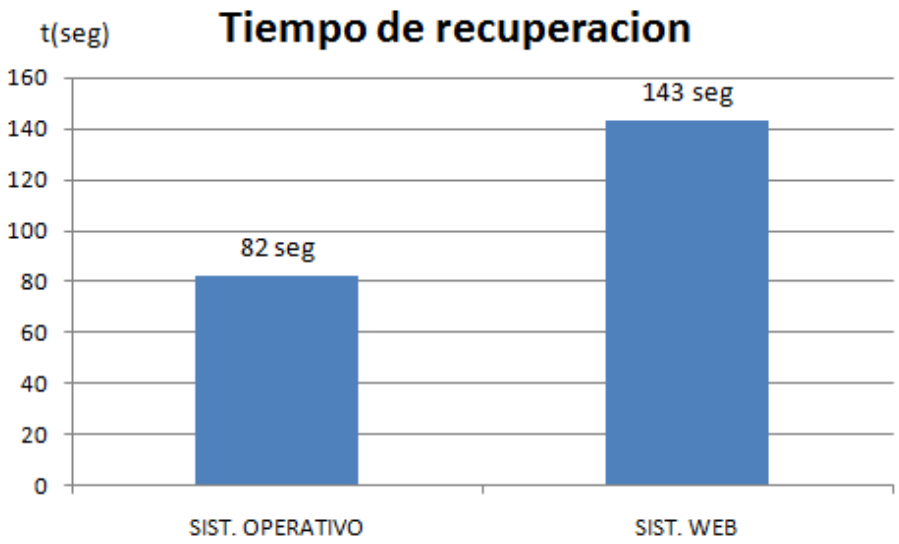

Figura 79: *Tiempo de recuperación del servidor*

Fuente*: Elaboración propia*

4.1.3. Pruebas de compatibilidad

Evalúa los siguientes aspectos:

- Compatibilidad de navegadores.
- Compatibilidad con diferentes dispositivos inteligentes.

a) Compatibilidad con navegadores.

Se realizó la prueba con los siguientes navegadores: Google Chrome Versión 68.0.3440.106, Internet Explorer versión 9.0, Mozilla Firefox versión 56.0 y Google Chrome para Android versión 68.0.3440.91.

El nivel de compatibilidad se resume en la tabla nro.14.

	CHROME	INTERNET EXPLORER	FIREFOX	CHROME ANDROID
TIEMPO DE DEMORA EN CARGAR INTERFAZ DE LOGIN	89 mseg	4seg	76 mseg	4seg
EJECUTA CODIGO ASP	SI	SI	SI	SI
EJECUTA JAVA SCRIPT	SI	SI	SI	SI
MUESTRA FORMATO LEGIBLE Y ORDENADO	SI	NO	SI	SI
REQUIERE ALGUN PAQUETE ADICIONAL PARA EJECUTAR LA APLICACIÓN	NO	NO	NO	NO

Tabla 13: *Compatibilidad entre navegadores*

Fuente*: Elaboración propia*

b) Compatibilidad con diferentes dispositivos inteligentes.-

Se realizó la simulación de funcionamiento en diferentes resoluciones utilizando el programa de testing **ScreenFly (http://quirktools.com/screenfly)**:

Pruebas en Smartphone

En la figura 80 se muestra la simulación con una resolución 360x640, se simula un smartphone del modelo Samsung Galaxy S5, las cajas de texto se redimensionan sin problemas.

***Figura 80**: Resolución de 360 x 640*

***Fuente**: Elaboración propia*

Pruebas en tablet

Se hace la simulación a una resolución de 768x1024, se simula una tablet del tipo Ipad , las cajas de texto y gráficos se redimensionan sin problemas (figura 81)

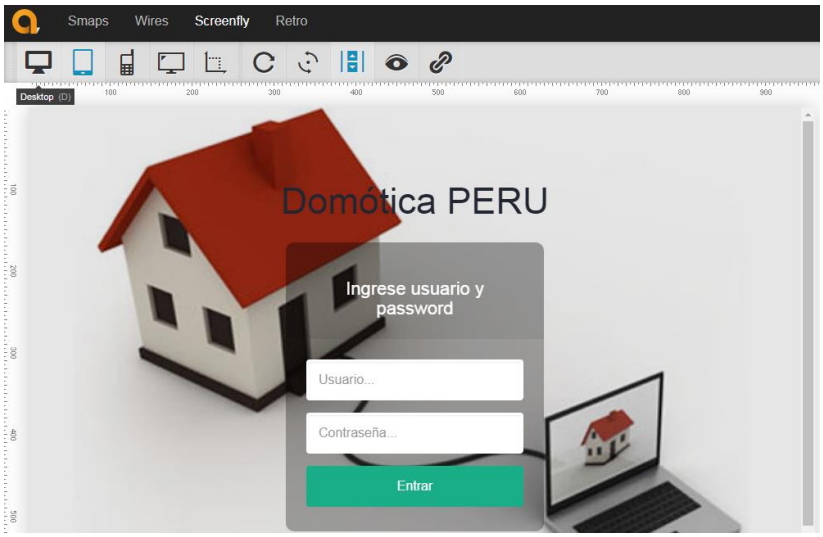

***Figura 81:** Resolución 768 x 1024*

***Fuente**: Elaboración propia*

CONCLUSIONES

1. Se implementó una estructura electrónica la cual permite la interacción entre el usuario, los sensores y actuadores mediante una interfaz Web logrando el control de los diferentes dispositivos domóticos.

2. Se desarrolló e implementó un sistema Web responsivo el cual se ajusta a cualquier dispositivo del tipo tablet o smartphone.

3. El prototipo propuesto servirá de ayuda al discapacitado para mejorar la usabilidad de los dispositivos de uso diario (focos, ventiladores, puertas, etc.).

RECOMENDACIONES Y MEJORAS AL PROYECTO

- Para el desarrollo a escala real del proyecto se recomienda la participación de diferentes profesionales tales como Ingenieros de Sistemas, Ingenieros electrónicos, Ingenieros Mecánicos, Ingenieros Eléctricos, Arquitectos.
- Se recomienda su implementación y puesta en producción en un asilo de ancianos, de esta manera se podría mejorar las actividades cotidianas de estas personas.
- Se recomienda el uso del sistema a personas que no tengan discapacidad total, quiere decir que su uso está orientado a personas en silla de ruedas que por lo menos puedan mover una mano y dispongan de sus capacidades cognitivas al 100%.
- Será necesario el montaje de un brazo metálico que sirva de apoyo a la tablet o dispositivo inteligente.
- Es recomendable que se pueda complementar el proyecto con un producto llamado MindWave que podría conectarse sin ningún problema al Arduino y lograr mover una silla de ruedas, y en general, manejar todas las opciones descritas del sistema con la mente (ondas cerebrales detectadas por el MindWave), de esta manera una persona paralítica podría usar el sistema.
- Para la autenticación del usuario e ingreso al sistema se podría usar un dispositivo biométrico de reconocimiento de huellas digitales, de esta forma solo bastaría con presionar su dedo en la pantalla de la tablet para poder ingresar al sistema, así se simplificaría el acceso y se daría más seguridad.
- Para la implementación del proyecto en un hogar donde se tenga distancias grandes entre el router y el dispositivo de control (tablet), se recomienda el uso de repetidores o Access Point que permitan extender la señal inalámbrica en todo el hogar.

BIBLIOGRAFÍA

- Arduino. (2018). *Arduino*. Obtenido de http://Arduino.cl/que-es-Arduino/
- ARDUINO, A. (2013). *APRENDIENDO ARDUINO*. Obtenido de https://aprendiendoArduino.wordpress.com/category/sensores/
- Bonino, D. (2011). *DOGeye Controlling your Home with Eye Interaction* , 30.
- Bonino, D. (2012). dWatch: a Personal Wrist Watch for Smart Environments. 8.
- Camargo, J. (2012). Reconocimiento devoz humana aplicado a la domotica. *IEEE* .
- Castellina, E. (2011). En *DOGeye Controlling your Home with Eye Interaction* (pág. 30).
- Castellina, E. (2014). Assistive Technology and Applications for the independent living of severe motor disabled users. 126.
- Chaparro, J. (2013). DOMÓTICA: LA MUTACIÓN DE LA VIVIENDA.
- *Cogaprel*. (2017). Obtenido de http://www.cogaprel.com/domotica.html
- DAVID, N. (2013). Design of a home automation system using Arduino.
- Eklund, D. (2013). A Home Automation Prototype.
- GONZALES, A. G. (23 de 01 de 2013). *panamahitek*. Obtenido de http://panamahitek.com/Arduino-mega-caracteristicas-capacidades-y-donde-conseguirlo-en-panama/
- Gonzales, R. (2006). *Conceptos de IIS*. Obtenido de http://rgonzales.blogspot.pe/2006/08/iis-concepto.html

- INEI. (2017). *INEI*. Obtenido de www.inei.gob.pe
- *Irisbond*. (2017). Obtenido de http://www.irisbond.com/home
- *La domótica en nuestra vida vía Definición ABC* . (s.f.). Obtenido de https://www.definicionabc.com/tecnologia/domotica.php
- LG. (2017). *LG*. Obtenido de http://www.lgtv.cl
- Llamas, L. (2016). *Medida de humedad de suelo con Arduino e higrómetro FC-28*. Obtenido de https://www.luisllamas.es/Arduino-humedad-suelo-fc-28/
- Mcrosoft. (2013). *Vsual Studio Community 2013*. Obtenido de https://msdn.microsoft.com/en-us/library/dn457348.aspx
- Microsoft. (2016). *Views and ViewModels* . Obtenido de https://docs.microsoft.com/en-us/aspnet/mvc/overview/older-versions/mvc-music-store/mvc-music-store-part-3
- *Movecare*. (2018). Obtenido de http://www.movecare-project.eu/
- Recuero, A. (2008). Estado actual y perspectivas de la domótica. 13.
- RIVERA, O. (2016). Control domótico por ondas cerebrales con apoyo mediante comandos de voz. 7.
- SCHOTT, M. R. (2005). investigación y desarrollo sistema prototipo de asistencia domótica para personas con movilidad limitada. 319.
- Smching. (2015). *Control de motores de corriente continua con Arduino* . Obtenido de http://www.instructables.com/id/L298N-Motor-Driver-Controller-Board/

- VIX. (2015). *Funcionamiento de control remoto*. Obtenido de https://www.vix.com/es/btg/curiosidades/2011/07/25/como-funciona-un-control-remoto

- Ziyu Lv, F. X. (2012). iCare: A Mobile Health Monitoring System for the Elderly .

- Calvo, F. (2009), análisis y diseño de una red domótica para viviendas sociales. 28.
- Roque, A. (2014), Diseño y desarrollo parcial de un sistema domótico para facilitar la movilidad de minusválidos. 132.
- Hornero, R. (2013), Diseño, desarrollo y evaluación de un sistema Brain Computer Interface (BCI) aplicado al control de dispositivos domóticos para mejorar la calidad de vida de las personas con grave discapacidad
- García, A. (2000), Nuevas tecnologías y personas con discapacidad. 295.
- Soumya, P. (2015)Design and Implementation of Home Automation System Using Facial Expressions. 616.
- López, J. (2016). Sistema domótico para mejorar el comodidad al realizar actividades para personas con discapacidad de locomoción utilizando tecnología Arduino y android. 76.
- Alarcon, A. (2016.)Estudio y diseño de un sistema domótico utilizando dispositivos móviles para mejorar la accesibilidad de las personas discapacitadas. 50.
- Isaza, J. (2011) Domotica y discapacidad. 3.