ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

Facultad de Ingeniería en Electricidad y Computación

"SISTEMA INTEGRAL PARA EL MONITOREO Y CONTROL DE LOS SONIDOS EN BARES Y ZONAS CÉNTRICAS DE LA CIUDAD EN TIEMPO REAL"

INFORME DE MATERIA INTEGRADORA

Previo a la obtención del Título de:

INGENIERO/A EN TELEMÁTICA

Presentado por:

JUAN MANUEL VILLACRESES MONTALVO

LISSETTE BEATRIZ MUNIZAGA SORIA

GUAYAQUIL – ECUADOR

AÑO: 2016

AGRADECIMIENTOS

Alcanzar la culminación de este proyecto ha sido gracias al apoyo y confianza de muchas personas que me han ayudado en todo el proceso de este trabajo, mi agradecimiento principal a Dios, y a mis padres que han sido mi soporte para poder continuar; brindándome siempre su amor y comprensión en este importante paso de mi vida.

Al Ing. Washington Velásquez por su aporte y colaboración dentro del desarrollo de este proyecto.

A mis hermanos Juan José, Ma. Luisa, Mauricio y Edinson que han sido mis ejemplos de superación, a los amigo(a)s que dentro del proceso no faltaron con una palabra de ánimo. A mis sobrino(a)s, mi abuelita Luisa Recalde, tío(a)s, primo(a)s y cuñado(a)s que de alguna manera se alegran con este triunfo alcanzado pensando en mi bienestar y el de mi hija.

A mi compañero Juan Villacreses por su paciencia brindada en duros momentos, Y no puedo dejar de agradecer a esas personas que Dios puso en mi camino cuando más lo necesité Ab. Holguer Ríos, Lcdo. Fernando Araujo, Sras. Daysi e Hilda Soria, Glenda Ruiz, Sres. Ronny Morán y Cristhian Lascano, Srta. Solange Jaime y en especial al Ing. Rory Gavilanes R.

Lissette Munizaga Soria.

A Mis padres, a quienes les agradezco su esfuerzo inmenso, su compañía y su motivación en todo instante en este camino que acabo de recorrer.

A mi novia quien estuvo siempre a mi lado a lo largo de todos estos años, siendo testigo de todos mis buenos y difíciles momentos, y siendo ese apoyo incondicional que se necesita para continuar.

No puedo dejar de agradecer a mis maestros que fueron parte fundamental para mi formación profesional.

Juan Manuel Villacreses Montalvo.

DEDICATORIA

Este logro lo dedico principalmente a mi hija Luciana Beatriz Delgado Munizaga porque es mi motor de lucha y superación diaria.

Hija mía a pesar de las muchas adversidades y duras pruebas que me toco atravesar dentro de mi vida, he alcanzado este logro por y para Ti; porque tu llegada a mi temprana edad no fue un fracaso en mí, al contrario me regalaste una vida más hermosa y alegre junto a TI; eres y serás mi eterna compañía por quien agotaré siempre mis últimas fuerzas y te defenderé ante cualquiera que intente lastimarte.

A mis padres Beatriz Soria Recalde y José Munizaga Arguello por el amor recibido dentro del hogar donde crecí, por hacer de mi infancia y vida la más feliz, por la formación y educación que me brindaron; porque confiaron en mí siempre y en esos momentos más difíciles lloraron conmigo, rieron conmigo; dándome siempre palabras de aliento en los momentos justos y recordándome que: “Nunca Dios pondrá en Mí, pruebas que no pueda superar” porque sé que mi vida sin ellos no puede ser igual. Por mis padres hoy soy quien soy.

Lissette Munizaga Soria.

A Dios por darme salud, sabiduría, perseverancia, inteligencia, paciencia y fuerzas para alcanzar esta meta tan importante en mi vida.

Existen fuerzas imparables, que siempre motivan a continuar. Nelly Cristina mi mayor inspiración para la culminación de este logro.

Juan Manuel Villacreses Montalvo.

TRIBUNAL DE EVALUACIÓN

Ing. Washington Velásquez	**Ing. Vladimir Sánchez**
PROFESOR EVALUADOR	PROFESOR EVALUADOR

DECLARACIÓN EXPRESA

"La responsabilidad y la autoría del contenido de este Trabajo de Titulación, nos corresponde exclusivamente; y damos nuestro consentimiento para que la ESPOL realice la comunicación pública de la obra por cualquier medio con el fin de promover la consulta, difusión y uso público de la producción intelectual"

.....

Lissette Munizaga Soria

.....

Juan Manuel Villacreses

RESUMEN

Se ha dado a conocer por la organización mundial de la salud, que los efectos causados por el ruido pueden ser perjudiciales para la salud de los humanos, además de ser incómodo para los mismos. En Ecuador se suscitan este tipo de molestias de manera muy seguidas en especial en ciudades muy habitadas, las leyes legislativas del país han tratado de proporcionar una ley para las personas causantes de estas incomodidades, pues al no tener un mecanismo para conseguir pruebas contundentes, existen distintos desacuerdos entre los legisladores. Luego de las investigaciones pertinentes realizadas entre los ciudadanos, se propone diseñar un sistema de monitoreo ambiental para de esta manera contribuir a las futuras decisiones de los legisladores, para el bienestar de los ciudadanos ecuatorianos previniendo un buen estado de salud y haciendo las ciudades más atractivas y tecnológicas.

La metodología utilizada fue indagar y obtener información acerca del problema presente, poniendo en práctica los conocimientos obtenidos dentro del proceso educativo universitario.

Se hizo uso de una tarjeta programable BeagleBone conectado respectivamente al sensor de ruido. Esta tarjeta es manipulada con una computadora, donde se pudo realizar la correcta programación de la misma; es necesario indicar que hace interconexión con una base de datos de MySQL, la sección programada por la tarjeta fue ejecutada en el lenguaje Python, mientras que el diseño y la programación de la página Web fue instanciada en HTML 5; fue necesario disponer de múltiples librerías para poder lograr los resultados deseados, proporcionando al usuario final un sistema dinámico, interactivo y de fácil uso para la obtención de datos requirentes y a su vez almacenados para lo que fue indispensable emplear un servidor Apache.

Los resultados obtenidos han sido muy favorables, se logró lo que se estimaba; el diseño y desarrollo de una Pagina Web donde el usuario registrado ingresa con su user_name y su contraseña incluidas en la base de datos, puede visualizar las

alarmas generadas por sectores cuando se ha sobrepasado de los límites permitidos de decibeles, puede configurar horarios establecidos como la ley lo indica para bares, locales comerciales, licoreras, entre otros.

Se generan mensajes de texto para notificar a los usuarios de las contravenciones adquiridas, inicialmente se le indica hasta por una segunda ocasión que ha incurrido en la sanción, a la tercera notificación se le informa de la misma manera mediante SMS al agente policial del UPC más cercano, para que proceda con las pruebas pertinente a la clausura del local, o sanciones futuras por el pleno de la Asamblea Nacional.

ÍNDICE GENERAL

AGRADECIMIENTOS II

DEDICATORIA III

TRIBUNAL DE EVALUACIÓN III

DECLARACIÓN EXPRESA IV

RESUMEN V

ÍNDICE GENERAL VII

CAPÍTULO 1 1

1. ANTECEDENTES Y JUSTIFICACIÓN 1

    1.1 Antecedentes 2

    1.2 Justificación 2

    1.3 Objetivo General 3

    1.4 Objetivos Específicos 3

    1.5 Alcances 4

    1.6 Limitaciones 4

CAPÍTULO 2 5

2. FUNDAMENTOS TEÓRICOS 5

    2.1 Lenguaje Python 5

    2.2 Django 5

    2.3 HTML5 (HyperText Markup Language) 6

    2.4 CSS3 (Cascading Style Sheet) 6

    2.5 JavaScript 7

    2.6 Jquery 7

2.7 Ajax 7
2.8 Flot 7
2.9 Grove (Sensor de Sonido) 7
2.10 Tarjeta Beaglebone Black 9
CAPÍTULO 3 11
3. IMPLEMENTACIÓN Y DESARROLLO 11
3.1 Desarrollo y Requerimiento 12
3.2 Manejo Electrónico: Bloque 1 12
3.2.1 Conexiones Sensor de Ruido – Tarjeta BeagleBone 12
3.3 Manejo Programable: Bloque 2 13
3.3.1 Codificación Sensor de Ruido 14
3.3.2 Python y MySQL 14
3.3.3 Desarrollo Web Python – Django 16
3.3.4 Servidor Apache 18
CAPÍTULO 4 19
4. ANÁLISIS DE RESULTADOS 19
4.1 Resultados Obtenidos 19
CONCLUSIONES Y RECOMENDACIONES 22
BIBLIOGRAFÍA 24
ANEXOS 26

CAPÍTULO 1

1. ANTECEDENTES Y JUSTIFICACIÓN

El ruido es la sensación auditiva inarticulada generalmente desagradable en el medio ambiente, se define como todo lo molesto para el oído, o más exactamente, como todo sonido no deseado. Según la Organización Mundial de la Salud, el nivel sonoro medio no debe exceder de los 45 dBA durante la noche, y 55 dBA durante el día [1]. Si hay demasiado ruido la salud se ve afectada; el ruido puede producir efectos sobre la salud de quienes sufren y por extensión, eso hace que la calidad de vida se resienta.

Problemas tales como: estrés, hipertensión, irritabilidad, falta de sueño, problemas de comunicación, interferencia de la comunicación hablada, perturbación del sueño, del descanso y de la relajación, impidiendo la concentración y el aprendizaje, tendencias a actividades agresivas, dificultades de observación, baja de rendimiento más frecuentemente en quienes padecen un exceso de ruido. La capacidad auditiva se deteriora causando hipoacusia (perdida de la misma) temporal o permanentemente en la banda comprendida entre los 75 dBA y 125 dBA, como se lo demuestra en la figura 1.1 donde indica los decibeles permitidos relacionados con el grado de hipoacusia.

Grado de hipoacusia y repercusión en la comunicación

Grado de hipoacusia	Umbral de audición	Déficit auditivo
Audición normal	0-25dB	
Hipoacusia leve	25-40dB	Dificultad en la conversación en voz baja o a distancia.
Hipoacusia moderada	40-55dB	Conversación posible a 1 o 1,5 metros.
Hipoacusia marcada	55-70dB	Requiere conversación en voz alta
Hipoacusia severa	70-90dB	Voz alta y a 30 cm.
Hipoacusia profunda	>90dB	Escucha sonidos muy fuertes, pero no puede utilizar los sonidos como medio de comunicación.

Figura 1.1: Decibeles permitidos relacionados con la hipoacusia [2]

1.1 Antecedentes

En nuestro país, el artículo 18 de la reforma a la Ley Orgánica de Prevención y Control de la Contaminación Ambiental establece que los niveles permisibles de ruido apto para las actividades humanas no deberán sobrepasar los 55 decibelios. [3]

Mientras que el numeral 20 norma las principales fuentes de contaminación acústica que pueden ser generadas por: vehículos, aeronaves, ferrocarriles, industriales, comerciales, discotecas, ruido doméstico, ruido social, maquinaria y equipo pesado, naves y maquinaria marítima. Por otro lado, en la exposición de motivos del informe se estipula que "lamentablemente este tema no ha sido dimensionado apropiadamente, convirtiéndose en un problema difícil de controlar, peor aún, no existe norma expresa que sancione esta abusiva práctica. La generación de ruido, por lo tanto, se ha convertido en un oprobioso comportamiento que menoscaba los derechos de las personas".

Los horarios y niveles de ruido establecidos en las leyes se mantienen de 06h00am a 20h00pm. En las áreas residenciales con un nivel máximo permitido de 55 dBA, en las áreas comerciales 60 dBA, sectores hospitalarios y educativos 45 dBA, y el área industrial 70 dBA; mientras que en los horarios de 20h00 hasta las 06h00, siguiendo el orden mencionado anteriormente de los sectores residenciales, comerciales, hospitalarios e industrial son: 40 dBA, 50 dBA, 35 dBA, 55 dBA. [3] . También fueron regularizados los horarios de funcionamiento de las distintas áreas mencionadas

- Lunes a Sábado de 17h00 a 03h00: Discotecas, Karaokes, bares, Night Club, Cantinas.
- Lunes a Sábado de 06h00 a 23h00: Restaurantes y Picanterías
- Lunes a Sábado de 12h00 a 22h00: Licorerías
- Lunes a Sábado de 06h00 a 22h00: Tienda de Abarrotes [4]

1.2 Justificación

Guayaquil es una ciudad altamente ruidosa, el problema está en que no hay sanción para las personas que lo ocasionan, ya que no existe un mecanismo para

realizar las necesarias mediciones del ruido dentro de las zonas céntricas y traficadas, así como los negocios nocturnos: bares, discotecas, night club de la ciudad.

Basados en informes y publicaciones de Diarios nacionales tal como: El Telégrafo en su artículo “El ruido de la ciudad puede provocar 100 enfermedades a los residentes” difundido en la fecha 03 de Agosto del 2014 [5]. Se da a conocer que dentro las leyes legislativas del país, aun no se ha encontrado una manera específica de poder sancionar a los causantes de tan graves molestias auditivas; existen distintas proposiciones entre los legisladores pero a su vez existen desacuerdos al no hallar prueba contundente para imponer dicha sanción.

Se propone realizar un controlador de ruido y sonidos dentro de las áreas previamente mencionadas, para así poder colaborar con las autoridades manejando informes de decibeles de ruido en tiempo real en zonas y establecimientos nocturnos, que maneje horarios establecidos por las autoridades competentes y que sea útil para un futuro ser manipulado como sustento y poder realizar las sanciones respectivas con pruebas justificadas.

1.3 Objetivo General

Implementar un Sistema de Monitoreo de Sonidos y Ruidos en bares y zonas céntricas de la ciudad mediante el uso de tecnología de hardware y software libre.

1.4 Objetivos Específicos

- Realizar informes a tiempo real de decibeles permitidos y sobrepasados de establecimientos nocturnos y zonas céntricas
- Desarrollar alertas para infractores de daños auditivos causados por ruido sobrepasado previo a la clausura o multa para el establecimiento.
- Desarrollar una página web adaptiva de fácil uso y acceso a los usuarios
- Implementar nodos de sensores sectorizados para el control de ruido en las localidades

- Emisión notificaciones GSM a los usuarios posteriormente a la alerta anunciada.

1.5 Alcances

Basado en los antecedentes y justificaciones previas al desarrollo e implementación del Sistema de Monitoreo de Ruido Ambiental, el alcance es dirigirlo a las organizaciones encargadas como son los Gobiernos Autónomos Descentralizados Municipales, Asamblea y Gobierno Nacional para colaborar en la continuidad del proceso de poseer ciudades inteligentes, precautelando el bienestar y buen vivir de los habitantes. Fomentar la ampliación de atractivos turísticos, así como cooperar con el crecimiento social y tecnológico al desarrollo del país, con la visión de pertenecer en el futuro a la lista de países de altas potencias mundiales.

1.6 Limitaciones

Las limitaciones analizadas son la falta de una eficiente red de cableado eléctrico para la correcta alimentación de los sensores, así como también que las zonas establecidas posean una excelente conexión a internet para efectuar el monitoreo constante; así mismo, un estudio geográfico de la perfecta ubicación de instalación de los sensores, para preservar su tiempo de vida útil de las diversas afectaciones climáticas como lluvia, sol, humedad que se expone en nuestro país.

CAPÍTULO 2

2. FUNDAMENTOS TEÓRICOS

La técnica de programación tiene como propósito la solución de problemas mediante el empleo de computadoras, la cual se identifica con los paradigmas de la programación estructurada, orientado a objetos y visual, resultando de gran importancia en la formación informática a través del desarrollo de interfaces para la conservación, tratamiento y transformación de la información; además con el desarrollo de programas se contribuye a la concepción científica del mundo, ya que la programación es un modelo y parte de la realidad objetiva, facilitando así el establecimiento de relaciones interdisciplinarias en cuanto al contenido de los problemas que se resuelven, potenciando el desarrollo de las habilidades de modelar, diseñar y algoritmizar.

El proyecto desarrollado es basado en diferentes lenguajes de programación, logrando obtener de ellos las mejores combinaciones y óptimos resultados.

2.1 Lenguaje Python

Python es un lenguaje de programación dinámico y fácil para su aprendizaje. Se maneja con estructuras de datos eficientes y de alto nivel con un enfoque simple por su versatilidad y rapidez de desarrollo, pero efectivo dentro de la programación orientada a objetos. La sintaxis de Python hace de éste un lenguaje ideal para el desarrollo rápido de aplicaciones en diversas áreas y sobre las diversas plataformas.

El intérprete de Python puede extenderse fácilmente con nuevas funcionalidades y tipos de datos implementados en C o C++, Java. La filosofía de Python es bastante análoga a la filosofía de Unix, es usado como un lenguaje de extensiones para aplicaciones personalizables. [6]

2.2 Django

Django es un WEB Framework; conjunto de componentes que ayuda a desarrollar aplicaciones gratuitas y open Source usado dentro del lenguaje

Python, compuesta de factores personalizables e intercambiables para el desarrollo de aplicaciones y sitios web; haciendo la programación para los desarrolladores más agradable, dinámica y sencilla en el proceso de trabajo. [7] Mantiene accesibilidad directa para tareas de programación frecuentes y convenciones claras sobre la forma de resolver problemas.

Se debe tener en cuenta varios requisitos que deben ser tomados en cuenta, al momento del uso de este framework tales como: dominio de la programación orientada a objetos, amplios conocimientos sobre patrones arquitectónicos y patrones de diseño. [8]

2.3 HTML5 (HyperText Markup Language)

HTML5 es la última versión del lenguaje HTML empleado para el desarrollo, estructura y contenido de páginas web compuesto por una serie de etiquetas, esta nueva versión contiene nuevos elementos, atributos y comportamientos que el navegador interpreta y da forma en la pantalla. [9]

Las nuevas características incluyen sintaxis <video>,<audio>, asegurando un excelente flujo de contenidos multimedia y diseños gráficos dentro de la página web, con las recientes actualizaciones realizadas es posible obtener elementos de formularios, mejoramiento de almacenamiento local, Obtención de la ubicación del usuario. [10]

2.4 CSS3 (Cascading Style Sheet)

Al referirse a CSS3 hablamos de un lenguaje para la presentación de documentación HTML, utilizada principalmente en la organización de la presentación y aspecto de una página web.

La filosofía de CSS3 se basa en separar lo que es el documento HTML de la presentación resultando de manera fácil cambiarle el aspecto a la Web, la integración entre HTML5 y CSS3 es vital para el desarrollo web, aunque oficialmente se trate de dos tecnologías completamente separadas. [11]

2.5 JavaScript

Es un lenguaje de secuencia de datos interpretado línea a línea que puede ejecutar comandos sin necesidad de compilación, con orientación a objetos; se encuentra incrustado directamente en las páginas HTML permitiendo a los desarrolladores el manejo e implementación de texto dinámico. [12]

2.6 Jquery

Es una librería de JavaScript óptima y rápida en sus funcionalidades. Mantiene un desempeño igual a HTML dentro de lo que es el recorrido y manipulación de los documentos, de eventos y animación. [13]

Aporta alguna de las siguientes ventajas:

- Provee un conjunto de funciones para animar el contenido de la página en forma muy sencilla.
- Hace transparente el soporte de la aplicación para los navegadores simples.

2.7 Ajax

Es una tecnología que tiene como función evitar las demoras propias existentes entre las peticiones del usuario y las respuestas del servidor mediante la transmisión de datos en segundo plano usando un protocolo especialmente diseñado para aquello. [14]

2.8 Flot

Es parte de las librerías de JavaScript, su rendimiento es muy eficiente y es accesible y compatible con todo tipo de gráficos; permite manejar cientos de puntos de datos con facilidad asociado con HTML5, su ejecución es accesible en los navegadores Firefox, Chrome, Internet Explorer, Safari y Opera. [15]

2.9 Grove (Sensor de Sonido)

Sensor que permite percibir, detectar y manejar la fuerza del sonido dentro del medio ambiente, su componente principal es un micrófono simple basado en un amplificador LM358 y un micrófono electret como se muestra en la figura 2.1. Su

producción es de señales analógicas, y admite integrarse fácilmente con los módulos lógicos existentes; entre sus distintas especificaciones destacamos las siguientes: [16] [17]

- Tensión de funcionamiento : 4 – 12V
- Corriente de Servicio (Vcc = 5V): 4Ma - 8Ma
- Ganancia de Voltaje (VS = 6V, f = kHz): 26 dB
- Sensibilidad de micrófono (1 kHz) : 58-48 dB
- Impedancia de micrófono: 2.2KΩ
- Micrófono de frecuencia: 16 – 20 kHz
- Microfono S/N ratio: 54 dB

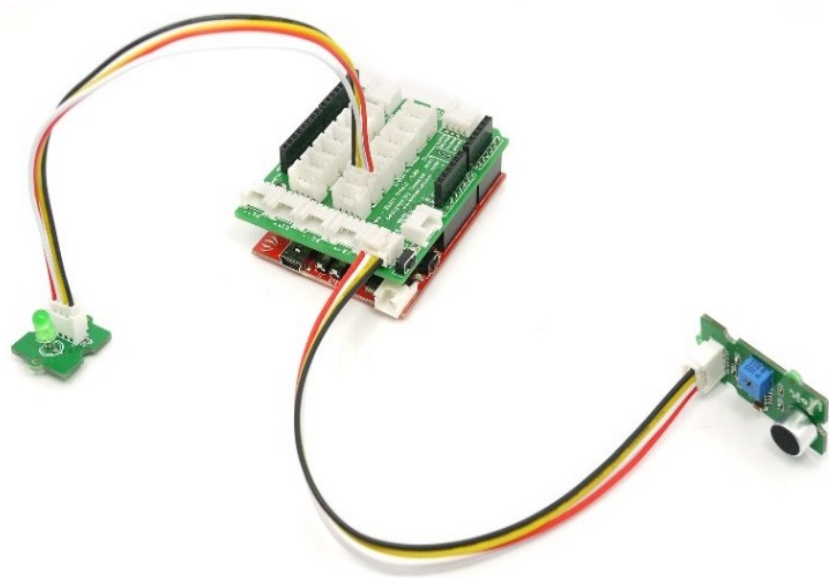


Figura 2.1: Sensor de Sonido – Grove [17]

Es posible usar el IDE Cloud9, como un poderoso editor de código manteniendo un espacio de trabajo en la nube; para la programación del sensor de ruido grove y la BeagleBone

2.10 Tarjeta Beaglebone Black

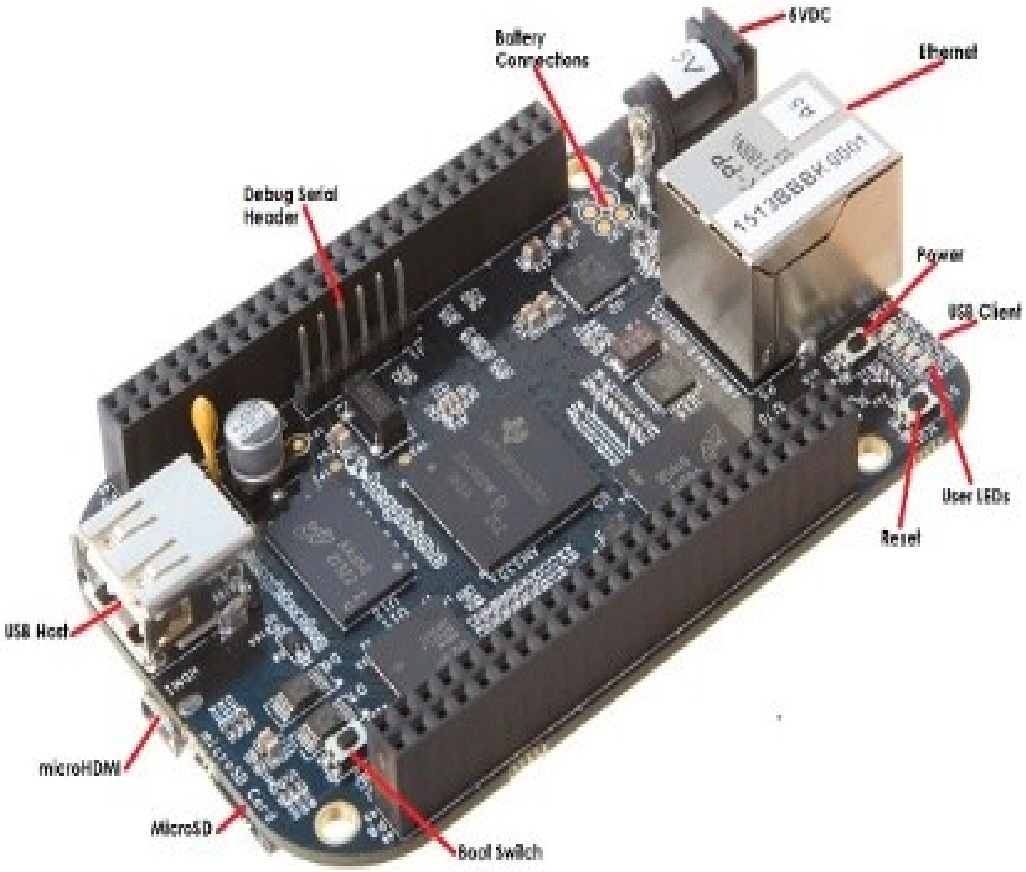


Figura 2.2: Modelo de tarjeta BeagleBone Black [18]

Es una placa digital programable muy pequeña en donde es posible ejecutar un sistema operativo, como puede ser LINUX/ANDROID; además cuenta con un microprocesador, memoria RAM, almacenamiento, puertos periféricos y otros elementos que tienen gran similitud con un computador como se muestra en la figura 2.2. El microprocesador de la tarjeta es un Sitara AM3358AZCZ que cuenta con un procesador ARM Cortex–A8, de un solo núcleo con un bus de 32 bits, permite trabajar con memorias RAM actuales como DDR2 y DDR3, contiene puertos Ethernet y puertos USB, también cuenta con una memoria DDR3L de 512 MB, esta memoria permite una velocidad de 1.32GB/S. El chip TP565217C recibe el voltaje de alimentación de la tarjeta y lo acopla a sus

demás partes como el procesador Sitara, los puertos y la memoria RAM [19]. En la figura 2.3 se muestran los datos técnicos de la tarjeta

	Feature	
Processor	Sitara AM3358BZCZ100 1GHz, 2000 MIPS	
Graphics Engine	SGX530 3D, 20M Polygons/S	
SDRAM Memory	512MB DDR3L 800MHZ	
Onboard Flash	4GB, 8bit Embedded MMC	
PMIC	TPS65217C PMIC regulator and one additional LDO.	
Debug Support	Optional Onboard 20-pin CTI JTAG, Serial Header	
Power Source	miniUSB USB or DC Jack	5VDC External Via Expansion Header
PCB	3.4" x 2.1"	6 layers
Indicators	1-Power, 2-Ethernet, 4-User Controllable LEDs	
HS USB 2.0 Client Port	Access to USB0, Client mode via miniUSB	
HS USB 2.0 Host Port	Access to USB1, Type A Socket, 500mA LS/FS/HS	
Serial Port	UART0 access via 6 pin 3.3V TTL Header. Header is populated	
Ethernet	10/100, RJ45	
SD/MMC Connector	microSD , 3.3V	
User Input	Reset Button Boot Button Power Button	
Video Out	16b HDMI, 1280x1024 (MAX) 1024x768,1280x720,1440x900 ,1920x1080@24Hz w/EDID Support	
Audio	Via HDMI Interface, Stereo	
Expansion Connectors	Power 5V, 3.3V , VDD_ADC(1.8V) 3.3V I/O on all signals McASP0, SPI1, I2C, GPIO(69 max), LCD, GPMC, MMC1, MMC2, 7 AIN(1.8V MAX), 4 Timers, 4 Serial Ports, CAN0, EHRPWM(0,2),XDMA Interrupt, Power button, Expansion Board ID (Up to 4 can be stacked)	
Weight	1.4 oz (39.68 grams)	
Power	Refer to Section 6.1.7	

Figura 2.3: Ficha técnica de Tarjeta Bleaglebone Black [20]

CAPÍTULO 3

3. IMPLEMENTACIÓN Y DESARROLLO

Se describe el proceso de instalación y programación entre los enlaces de la tarjeta programable, sensores de ruido, módulo de envío y recepción de mensajes gsm, router, Base datos y pagina web.

La página web recepta mediante la base de datos la información captada por el sensor de ruido y lo presenta en un gráfico de graduaciones los altos y bajos niveles de decibeles obtenidos.

En el momento preciso cuando se identifica un sonido o ruido que ha excedido del rango permitido en los horarios establecidos, automáticamente se emite una alerta mediante mensaje de texto al celular del usuario final, siendo este primeramente el dueño del establecimiento que presentó dicha infracción; esta notificación se registra en la base de datos con la información requerida: número de establecimiento, dueño del mismo, hora, fecha de la infracción y emisión de dicho mensaje. En caso de que el establecimiento llegue a recurrir por una tercera ocasión en la contravención, además de alertar al dueño, se extiende la misma notificación a algunos de los agentes policiales encargados del UPC (Unidad de Policía Comunitaria) más cercano previamente registrados con sus números de contactos. El proceso se muestra en la figura 3.1

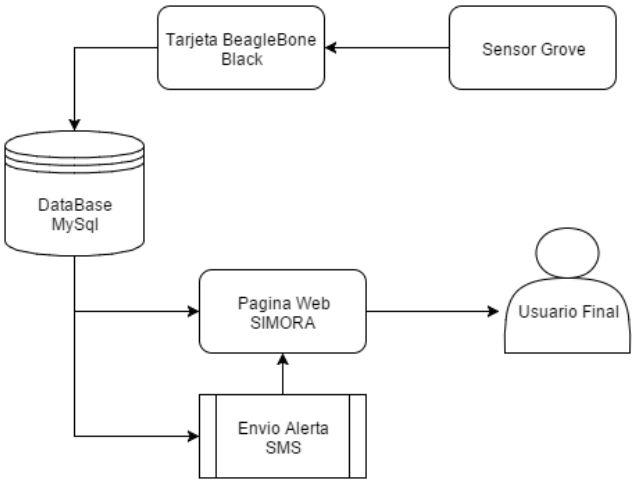


Figura 3.1: Diseño Esquemático de Sistema y Control de Monitoreo

3.1 Desarrollo y Requerimiento

El principal servicio que se presenta está enfocado en la información que se transmite desde el sensor de ruido, captando los decibeles ambientales hacia los usuarios finales mediante la generación de un reporte respectivo, para lo cual el desarrollo ha sido seccionado y realizado en dos bloques:

- **Bloque 1 :** Manejo Electrónico
- **Bloque 2 :** Programación de dispositivos, Control y Almacenamiento de Base de Datos, Envío de Alertas GSM, Desarrollo de Pagina Web

3.2 Manejo Electrónico: Bloque 1

La Tarjeta BeagleBone es posible encenderla por medio de dos fuentes de alimentación, usando el puerto miniUSB hacia la computadora; es indispensable recalcar que al utilizar esta conexión la placa emitirá 0.5 Amperio (A); puesto que en el diseño e infraestructura es necesario el uso de varios sensores (2 sensores de ruido, 1 módulo GSM) la cantidad de corriente emitida bajo esta conexión no es suficiente, se recurrió a la segunda conexión de la tarjeta bajo su Jack DC que transmite 5DVC.

Es importante indicar que no se tuvo la obligación de realizar la instalación de Drivers en la BleagleBone Black; ya que la misma viene con el sistema operativo DEBIAN [21]

3.2.1 Conexiones Sensor de Ruido – Tarjeta BeagleBone

- Salida Analógica: El pin macho de salida analógico del sensor de ruido puede ir conectado a los pines hembras **33, 35, 36, 37, 38, 39, 40** de la tarjeta, que son los designados para los puertos analógicos (AIN – Analog Input)
- Tierra Física: El pin macho de Tierra Analógica del sensor, va conectado al pin hembra **34** de BeagleBone designada para el GND (ground)
- Alimentación Positiva de Tensión: El pin macho VCC+ del Sensor de Ruido, podría ir unido a los pines hembra **3, 4, 5, 6** que

corresponden a la entrada de tensión positiva en la tarjeta, pero es importante indicar que los pines **3, 4** manejan únicamente un voltaje de 3DVC; más el sensor requiere una alimentación de 5DVC y por este motivo se escoge los pines **5** o **6** que manejan dicha voltaje. Se muestra en la figura 3.2 todos los pines habilitados para cada función de la tarjeta.

Como se hizo uso de 2 sensores de ruido y es necesario su conexión a tierra física (ground) y la tarjeta programable solo posee un pin hembra de Tierra Analógica se construyó un puente para que ambos vayan conectados.

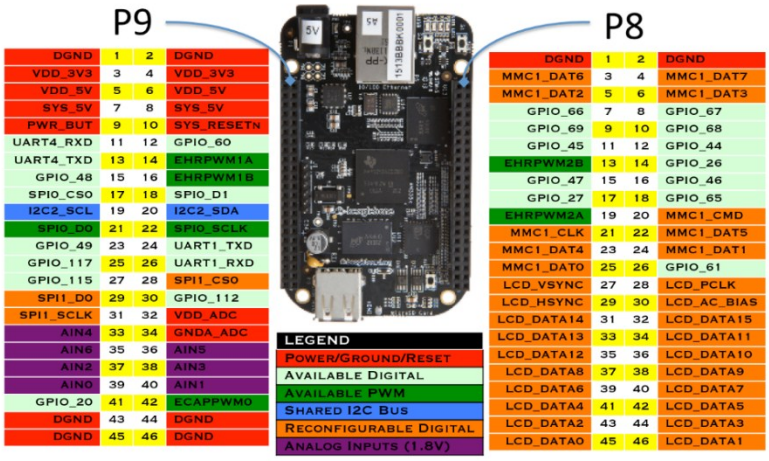


Figura 3.2: Diseño de Entrada y Salidas de pines en BeagleBone Black [22]

3.3 Manejo Programable: Bloque 2

Dentro el bloque programable es necesario el uso de la computadora, para lograr el enlace entre el manejo electrónico descrito previamente en el bloque anterior y la tarjeta BeagleBone, y al mismo tiempo obtener la conexión existente desde la base de datos hacia la página web desarrollada en el lenguaje Python.

Se realizó el correcto algoritmo y su posterior programación para controlar y almacenar los datos obtenidos desde el sensor de ruido, así como también la debida proyección para poder emitir las notificaciones GSM.

3.3.1 Codificación Sensor de Ruido

El código principal del Sensor de ruido encargado de manejar el tiempo a esperar por sonidos en segundos, programación de las potencias y voltajes del sensor, haciendo las necesarias llamadas e invocaciones de funciones para enviar la señal receptada a la base de datos.(Ver anexo 1)

3.3.2 Python y MySQL

El lenguaje python, independientemente de la base de datos con la que se lo manipule, sus métodos y procesos de conexión, lectura y escritura de datos siempre serán los mismos. En particular se usó como base MySQL, para lo cual se debió trabajar con el módulo característico *MySQLdb*. A diferencia de otros módulos existentes de la librería estándar de Python, este fue instalado manualmente.

Para ello, se ejecutó el siguiente comando:

```
sudo apt-get install Python-mysqldb
```

Para poder exponer la base de datos se ha recopilado la información necesaria, esto denota que se realizó un estudio previo junto con la ciudanía acerca del tema, de los problemas ocasionados por el exceso de ruido en sus sectores, investigaciones previas para determinar de esta manera lo que ellos hacen, desean y así poseer una idea más clara de lo que se necesitaba. Se hizo la respectiva identificación de los objetos y entidades más importantes que vayan a ser administrados como se muestra en la figura 3.3, una vez establecidos los candidatos para las tablas, identificamos los tipos de información para cada uno de ellos. Las tablas que se han determinado para el desarrollo de esta base de datos son las siguientes.

- *Usuarios:* Permite la creación, acceso y verificación de usuarios nuevos y existentes
- *Alarma por Defecto:* Al ser instalada una nueva alarma en un sector previamente sin especificar siendo este hospitalario,

Industrial, Residencial, Centros nocturnos; manejara rangos de decibeles por defectos

- *Programar Alarma:* Manipula alarmas en casos excepcionales donde se admiten horarios diferentes a los ya establecidos, siendo estos feriados y elecciones electorales
- *Alarma Detectada:* Facilita guardar la alarma generada
- *Control:* Se rige a los parámetros de control establecidos tales como: intervalo de alarma, milivoltios de Referencia, decibeles de referencia.
- *Localidad:* Opera en los sectores que se disponen y se detallen
- *Supervisor:* Por cada localidad y usuario, se fijara un Usuario Supervisor, a quien se le emitirá las notificaciones GSM, y alertas correspondientes.

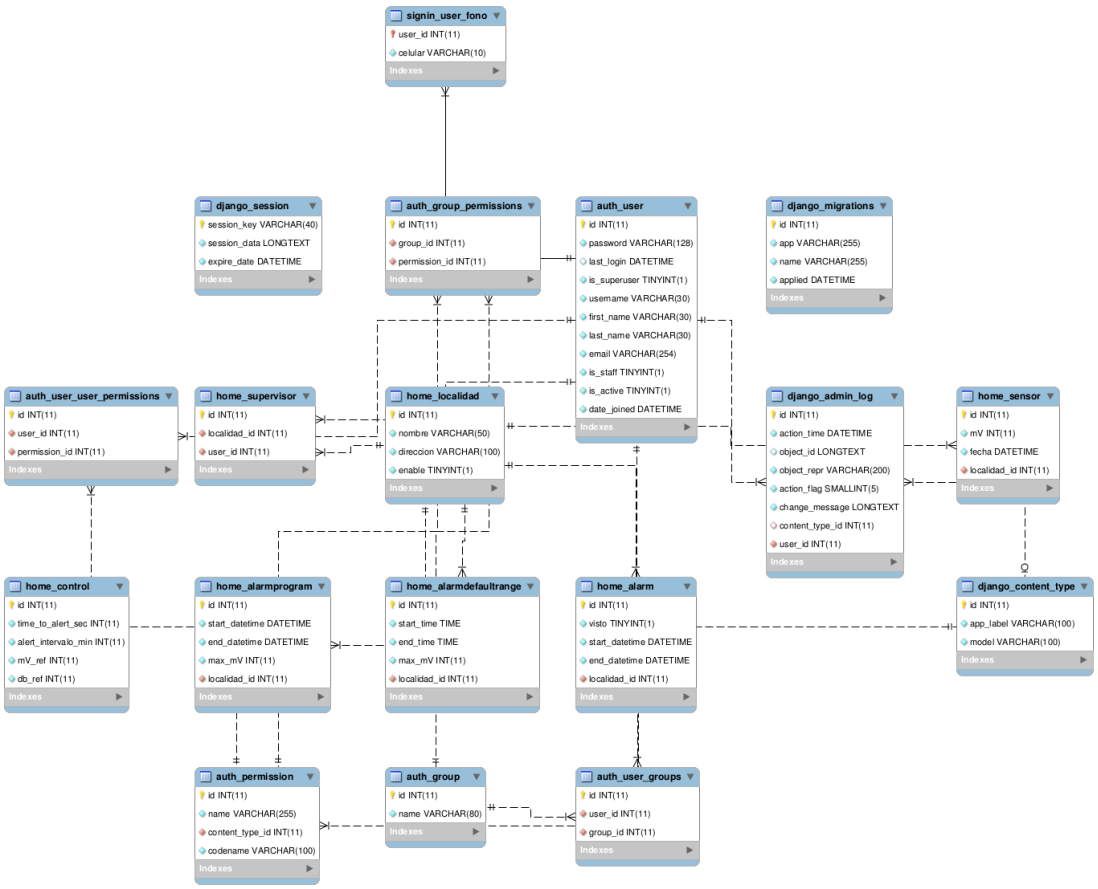


Figura 3.3: Modelo Conceptual de la Base de Datos

Para conectarnos a la base de datos y ejecutar cualquier consulta, el procedimiento realizado es:

1. Abrir la conexión y crear un puntero
2. Ejecutar la consulta
3. Traer los resultados; si se aplica de una selección o Hacer la escritura; cuando se elimina, actualiza o insertan datos.
4. Cerrar el puntero y la conexión

Una forma simple de acceder a la base de datos es ejecutando algunas líneas de código. (Ver anexo 2)

Entre las funciones más utilizadas de la placa BeagleBone y programadas en el lenguaje de Python son: insertar datos, envío de notificaciones mediante correo electrónico, envío de notificaciones mediante GSM. *(*Ver anexo 3)

3.3.3 Desarrollo Web Python – Django

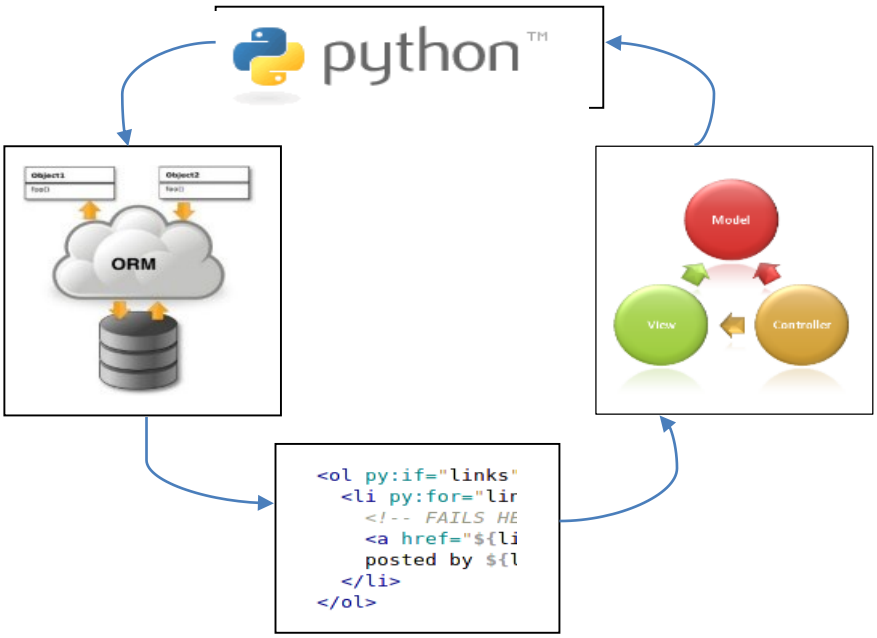


Figura 3.4: Diseño Esquemático de página web basado en elementos conceptuales.

En el desarrollo de la página web se ha basado en el elemento conceptual de *Python,* también se ha definido un *"Mapeado Objeto Relación (ORM)"* que es una técnica de programación que permite acceder a las bases de datos definiendo objetos y llamando a sus métodos.

El proceso como se muestra en la figura 3.4 cuenta con un *sistema de plantillas* que procesa los archivos, donde se combina HTML con la lógica de programación de Python para hacer dinámico al sitio web, y se concluye con el *Modelo-Vista-Controlador (MVC)* que es un patrón de arquitectura de software que divide las aplicaciones en tres capas fundamentales, *Modelo* (el código que accede únicamente a la base de datos), *Vistas* (las funciones que están relacionadas con la interfaz gráfica) y *Controlador* (la lógica interna de la aplicación, que une al modelo con la vista)

El uso del framework web Django de Python ha proporcionado a la implementación y desarrollo, construir la aplicación de nombre *SIMORA (Sistema de Monitoreo de Ruido Ambiental)* de una manera más rápida y con menos código. La función *view (vista)* de Python se encarga de tomar una petición web y devuelve una respuesta del mismo tipo, siendo estas: el contenido HTML de la página creada, una redirección URL, error 404, documento XML o Jason Response. En esta función se almacena el contexto (diccionario) de Python que será reenderizado en las plantillas.

Se manejan diversas vistas, se destaca algunas de las más utilizadas. (Ver anexo 4)

Entre los contextos de las vistas se detallan:

- *User_FullName:* Indica el nombre completo del usuario que ha sido loggeado y mantiene sesión activa
- *User_Last_Login:* Presenta la última fecha y hora de ingreso del Usuario que se encuentra activo
- *Localidades:* Sectores donde se harán los respectivos controles y emisiones de alertas

- *Noti_Number:* Proporciona un número indicador y contador de las alertas emitidas que aún no han sido observadas.

Se especifica que dentro del proceso de desarrollo en el área programable se hizo uso de la técnica para crear aplicaciones web interactivas como lo hace *AJAX,* el usuario realiza la petición mediante la página web, mientras se mantiene y se realiza la comunicación asincrónica con el servidor en un segundo plano sin interferir con la visualización ni el comportamiento de la página. Estas funciones de llamadas de AJAX se efectúan en el lenguaje interpretado de JAVASCRIPT, se destaca las funciones mayormente usadas. (Ver anexo 5)

Como se ha dado a conocer en el capítulo anterior la librería FLOT es la que se utiliza para la realización de los gráficos. (Ver anexo 6)

3.3.4 Servidor Apache

Se consideró y se escogió un servidor de código abierto para enviar páginas web estáticas y dinámicas, que sea multiplataforma, modulares, extensibles y populares, en el cual se pudo implementar el Sistema de Ruido Ambiental que vaya almacenando la información. Se detalla las configuraciones principales del servidor. (Ver anexo 7)

CAPÍTULO 4

4. ANÁLISIS DE RESULTADOS

A lo largo de todo el proceso de implementación se tuvo como idea principal establecer un *sistema* que permita censar los niveles de ruido dentro de la ciudad ocasionados por los diversos factores; en algún momento se planeó como sanción a los infractores cuando haya incurrido por una tercera ocasión, interrumpir la conexión eléctrica del local remotamente pero luego del análisis, se decidió no realizarla ya que podría ocasionar daños a los artefactos electrodomésticos de dicho lugar e incluso llegar a mantener contrariedades legales con los mismos. Entonces se logró legitimar un sistema con algunos funcionamientos que ayudaran a los usuarios tanto administradores, como autoridades pertinentes.

4.1 Resultados Obtenidos

En las capturas de pantalla realizadas se observan los resultados obtenidos de la aplicación habiendo realizado pruebas pertinentes, simulando como sector el barrio de Las Peñas, antes de todo se da una pequeña introducción al usuario para el uso del mismo.

Para poder acceder al Sistema de monitoreo de ruido ambiental (SIMORA) es necesario ingresar al navegador de su preferencia siendo este Google Chrome, Internet Explorer, Mozilla entre otros y digitalizar la siguiente dirección *www.tesis.dynu.com* y realizar la correcta validación de usuarios. (Ver anexo 8*)*

El usuario previamente registrado y almacenado en la base de datos con su user_name y su contraseña ingresa haciendo la correcta digitación de los mismos.

Si el usuario olvida su contraseña, la página le ofrece la opción de recuperación de contraseña (¿Olvidaste la contraseña?), que re direcciona a una vista donde solicita ingresar la dirección de correo guardado y así se le emite la información pertinente para poder reestablecerla. (Ver anexo 9)

Una vez que el usuario haya validado sus datos, se presenta la pantalla principal del sistema con sus pluggins y sus viñetas de despliegues. (Ver anexo 10)

El usuario puede observar el monitoreo en tiempo real, del lado izquierdo de la misma se denota las opciones despegables alarmas, reportes, Zonas – Gráficos, Administración. Mientras que del lado superior derecho se observa un icono con un pequeño número, este indica las notificaciones que aún no han sido observadas por el usuario, a la vez se contempla el nombre completo del usuario ingresado y la última fecha y hora de su ingreso al sistema. (Ver anexo 11)

Es importante recalcar que se encuentra la opción de Perfil, la que permite al usuario personalizar sus datos dinámicamente para poder interactuar entre los demás; desde modificar sus credenciales hasta introducir una foto que pueda ser vista por las personas cuando ingresen a su perfil, se señala que en este mismo procedimiento se halla la opción de *Salir*, que al ejecutarla automáticamente se realiza el cierre de la sesión activa. (Ver anexo 12)

Entre las principales funcionalidades que demanda el sistema creado es la muestra de las alarmas archivadas, como resultado de esto, se realizó una prueba en el sector de las Peñas, como se observa en la figura 4.1 se enciende la respectiva luz cuando se cometió la infracción en la localidad previamente mencionada.

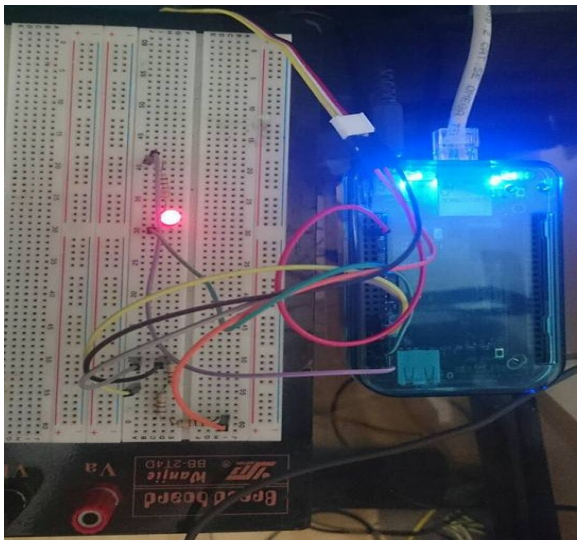


Figura 4.1: Muestra de infracción observada en “Las Peñas”

En la presentación de SIMORA en la ventana de alarmas se logra visualizar la localidad, en el caso analizado de “Las Peñas”; la dirección exacta dentro de donde fue emitida la alarma, muestra su fecha de inicio y su fecha de fin (ver anexo 13) y a la vez permite observar el grafico de la misma. (Ver anexo 14)

El reporte de las alarmas generadas es presentado en un tipo de archivo PDF y muestra además los milivoltios alcanzados, permitiendo visualizar los parámetros establecidos de milivoltios, decibeles de referencia, tiempo máximo de ruido, intervalo por alarma; una vez esclarecidos estos parámetros las alertas se darán cuando excedan de los mismos. Además se detallan las horas de inicio y de fin que se puede mantener ruido en las “Las Peñas”, esta opción es muy útil cuando las autoridades han establecido ya un rango de horas y en feriados aumentan las mismas o disminuyen como sucede en elecciones presidenciales. Todos los parámetros previamente mencionados, solo pueden ser insertados, modificados u eliminados con la opción del usuario Administrador.

CONCLUSIONES Y RECOMENDACIONES

Conclusiones

Una vez finalizado la implementación del proyecto y con los resultados obtenidos se pueden realizar las conclusiones que darán un enfoque global de la ejecución de la aplicación.

1. Se logró presentar a tiempo real los niveles de decibeles permitidos dentro de las distintas zonas y sectores de la ciudad, dando a conocer a los infractores su participación dentro de la misma.
2. Se generan reportes con tipo de archivos PDF y Excel para contribuir a las respectivas pruebas en contra de los infractores y tomar acciones hacia los mismos.
3. Se detectó cuáles son las necesidades reales de las personas que habitan y militan con los molestosos problemas de ruido; los procesos operativos del sistema se apegan a la realidad de la convivencias de estas personas, definiendo de manera clara y tangible los beneficios de salud y del buen vivir.

Recomendaciones

En todo el proceso de implementación y ejecución del proyecto surgieron ciertos contratiempos o problemas que fueron resueltos uno a uno, por lo que se redacta a continuación recomendaciones necesarias para una ejecución exitosa o con menos probabilidades de errores.

1. Instalar en la placa el paquete NTP para mantener siempre la horas sincronizas mundiales
2. Realizar una instalación limpia de Debian con su última imagen actualizada en la página del fabricante solo eligiendo los paquetes necesarios para la implementación para evitar el consumo de recursos de memoria y CPU
3. Previamente informarse con la ficha técnica de la tarjeta y del sensor para evitar futuros daños en los mismos.

4. No exceder de tres sensores de ruido a la placa debido a que cada sensor consume el 25% de la capacidad del procesador, ya que el muestreo de cada sensor se encuentra en su máximo nivel.
5. El tipo de sensor debe constar de una salida analógica y que opere con las frecuencias que puede receptar el oído humano
6. Al realizar el uso de módulo GSM verificar que opere dentro de las bandas usadas en el país

BIBLIOGRAFÍA

[1] Diba, (2015,octubre). Organizacion Mundial de la Salud [online]. Disponible en: https://www.diba.cat/c/document_library

[2] Sites.google, (2015,octubre). Decibeles relacionados con la hipoacusia. [online]. Disponible en: https://sites.google.com/site/tecnologiadeaudio208030g141

[3] Asamblea Nacional, (2015,octubre). Ley de control de la contaminacion ambiental [online]. Disponible en: http://ppless.asambleanacional.gob.ec

[4] Ministerio interior, (2016,enero). Horarios permitidos a establecimientos [online]. Disponible en: http://www.ministeriointerior.gob.ec

[5] El Telegrafo, (2015,octubre). Publicacion de Enfermedades a causa del ruido [online]. Disponible en: http://www.telegrafo.com.ec/noticias

[6] R. G. Duque, Python *para todos*, XKEITHFUL, 2013.

[7] Django Book, (2015,octubre). Framework Django [online]. Disponible en: http://www.djangobook.com/en/2.0/chapter01.htm

[8] Lsi universidad española, (2015,diciembre). Django – Python [online]. Disponible en: www.lsi.us.es

[9] B. LAWSON, Desarrollo web, Introducing HTML 5, Copyrighted Material, 2010.

[10] Latin designer, (2015,noviembre). Caracteristicas HTML 5 [online]. Disponible en: http://latindesigner.net

[11] Z. M. Gillenwater, Adegiusti, Stunning CSS3, Copyrithed Material, 2011.

[12] S. Stefanov, Javascript, JavaScript Patterns, O'REILLY, 2010.

[13] D. S. McFarland, Javascriptya, Javascrip & Jquery 3rd edition, O'REILLY, 2009.

[14] Api Jquery, (2015,diciembre). Tecnologia Ajax [online]. Disponible en: http://api.jquery.com/jquery.ajax

[15] Jqueryflottutorial, (2015,octubre). Libreria Flot [online]. Disponible en: http://www.jqueryflottutorial.com

[16] L. A. Hernandez, Didacticaselectronicas, Programacion Orientada o Objetos, UNAM, 2006.

[17] Seeedstudio, (2015,noviembre). Sensor de Ruido Grove [online]. Disponible en: http://www.seeedstudio.com/wiki/Grove_-_Sound_Sensor

[18] Electronilab, (2016,enero). Tarjeta programable BeagleBone [online]. Disponible en: http://electronilab.co/tienda/beaglebone-black-rev-c-arm-cortex-a8/.

[19] Digikey, (2015,diciembre). Sistema operativo Sitara [online]. Disponible en: http://www.digikey.com/product-detail/es

[20] Elinux, (2016,enero). Ficha Tecnica BeagleBone [online]. Disponible en: http://elinux.org/Beagleboard

[21] Debian, (2016,febrero). Sistema Operativo Debian [online]. Disponible en: https://www.debian.org

[22] BeagleBoard, (2015,octubre). Puertos de conexion BeagleBone [online]. Disponible en: http://beagleboard.org

ANEXOS

Anexo 1 (Codificación Principal de la BeagleBone)

```
from classes.db import MyDB
import Adafruit_BBIO.ADC as ADC
import Adafruit_BBIO.GPIO as GPIO
from time import sleep
import datetime
import MySQLdb
from email_alert import alert_email
import thread
import time
from sms import send_sms
def sensor (led_pin, analog_pin, loc_id ):
db= MyDB()
# led = "P9_42"
led = led_pin
GPIO.setup(led, GPIO.OUT)
fecha = None
fecha_high= None

time_wait = 4
ADC.setup()
analogPin=analog_pin
while(1):
signalMax = 0
signalMin = 1800
print "while"
for index in range(10):
potVolt=ADC.read_raw(analogPin)
if potVolt > signalMax:
signalMax = potVolt
elif potVolt < signalMin:
signalMin = potVolt
peakToPeak = (signalMax - signalMin)+100;
db.set_mV(peakToPeak,loc_id, datetime.datetime.now())
if GPIO.input(led):
if peakToPeak < db.get_mV(loc_id):
if(datetime.datetime.now()-fecha_high).total_seconds()time_wait:
GPIO.output(led, GPIO.LOW)
else:
fecha_high = datetime.datetime.now()
if(datetime.datetime.now()-fecha).total_seconds()db.max_time() :
if(datetime.datetime.now() - db.last_alarm(loc_id)).total_seconds()/60 >
db.alarm_interval():
db.set_alarm(fecha,datetime.datetime.now(),loc_id) thread.start_new_thread(
send_sms, (db.get_list_phones(loc_id),db.get_loc_name(loc_id),
datetime.datetime.now()) )
                        thread.start_new_thread( alert_email,
(db.get_list_email(loc_id),db.get_loc_name(loc_id), datetime.datetime.now())
)
GPIO.output(led, GPIO.LOW)# send_sms()
else:
if peakToPeak > db.get_mV(loc_id):
GPIO.output(led, GPIO.HIGH)
fecha = datetime.datetime.now()
fecha_high = fecha
```

Anexo 2 (Código de acceso a la base de Datos)

```
import MySQLdb
import datetime
class MyDB(object):
db_connection = None
db_cur = None
def __init__(self):
self._db_connection = MySQLdb.connect('192.168.1.137','casa','casa','simora'
)
self._db_cur = self._db_connection.cursor()
def set_mV(self, mV, loc_id, fecha):
q = "INSERT INTO home_sensor(mV,fecha, localidad_id) VALUES (%s,'%s',%s)" %
(mV, fecha, loc_id)
self._db_cur.execute(q)
self._db_connection.commit()
def set_alarm(self, start_datetime,end_datetime, loc_id):
q = "INSERT INTO home_alarm (visto, start_datetime, end_datetime,
localidad_id) VALUES (0,'%s','%s',%s)" % (start_datetime, end_datetime,
loc_id)self._db_cur.execute(q)
```

Anexo 3 (Código de funciones programadas en lenguaje Python)

- ***Insertar Datos***

```
import Adafruit_BBIO.ADC as ADC
import MySQLdb
db = MySQLdb.connect("192.168.1.137","root","bbb","tesis_db" )
cursor = db.cursor()
ADC.setup()
analogPin="AIN5"
while(1):
signalMax = 0
signalMin = 1800
for index in range(10):
potVolt=ADC.read_raw(analogPin)
if potVolt > signalMax:
signalMax = potVolt
elif potVolt < signalMin:
signalMin = potVolt
peakToPeak = (signalMax - signalMin)+100;
sql_insert = "INSERT INTO home_sensor(mV,fecha, localidad_id) VALUES
(%s,now(),1)" % peakToPeak
try:
# Execute the SQL command
cursor.execute(sql_insert)
# Commit your changes in the database
db.commit()
except:
# Rollback in case there is any error
db.rollback()
```

- ***Envío de notificaciones mediante correo electrónico***

```
import smtplib
from email.mime.multipart import MIMEMultipart
from email.mime.text import MIMEText
def alert_email():
fromaddr = "juan_men10@hotmail.com"
toaddr = ["juanx90@msn.com", "jmvillac@espol.edu.ec"]
msg = MIMEMultipart()
msg['From'] = fromaddr
msg['To'] = ", ".join(toaddr)
msg['Subject'] = "SUBJECT OF THE MAIL"
body = "YOUR MESSAGE HERE"
msg.attach(MIMEText(body, 'plain'))
server = smtplib.SMTP('smtp-mail.outlook.com', 25)
server.starttls()
server.login(fromaddr, "contrasena")
text = msg.as_string()
server.sendmail(fromaddr, toaddr, text)
server.quit()
```

- ***Envío de notificaciones mediante GSM***

```
from twilio.rest import TwilioRestClient
# Your Account Sid and Auth Token from twilio.com/user/account
account_sid = "ACd48d7cada5149244f0abda0a4c002e68"
auth_token  = "43a2a3079d2f2d26809ccb0951f4c6c0"
client = TwilioRestClient(account_sid, auth_token)
message = client.messages.create(body = "Ha recibido una alerta del Sistema
de Control SIMORA”,
to="+593968984786",    # número telefónico de administrador SIMORA
from_= "+593986245176") # número telefónico del infractor
print message.sid
```

Anexo 4 (Código Django de vistas más utilizadas)

- ***Login_View***

```
@csrf_exempt
def login_view(request): form = Login()
if not request.user.is_authenticated():
if request.method == "POST":
form = Login(request.POST)
if form.is_valid():
username = form.getUsername()
password = form.getPassword()
user = authenticate(username=username, password=password, is_active=True)
if user is not None:
login(request, user)
return HttpResponseRedirect(reverse('home'))
else:
messages.error(request, u'Ha ingresado un usuario o contraseña
incorrecta')
return render_to_response("login.html", {"form": form},
context_instance=RequestContext(request))
else:
```

```
messages.error(request, u"Por favor para iniciar sesión debe ingresar su
usuario y su clave.")
return render_to_response("login.html", {"form": form},
context_instance=RequestContext(request))
else:
print "fuera del post"
else:
return HttpResponseRedirect(reverse('home'))
return render_to_response("login.html", {"form": form},
context_instance=RequestContext(request))
```

- ***Home View***

```
@login_required(login_url='/')
def home(request):
if not request.user.get_full_name()=="":
user = request.user.get_full_name()
else:user = request.user.get_username()
user_last_login = request.user.last_login
print user
for e in localidad.objects.all():
print e.get_nombre()
context = {
'user_fullname': user,
'user_last_login': user_last_login,
'localidades': localidad.objects.all(),
'noti_number':AlarmDetected.objects.filter(visto=False).count(),
}
return
render_to_response("home.html",context,context_instance=RequestContext(req
uest))
```

Anexo 5 (Código de funciones de llamadas en AJAX)

- ***Envía datos a página web y grafica en tiempo real***

```
@csrf_exempt
@login_required(login_url='/')
def ajax(request):
data = []
loc_name = request.GET.get('zone')
# print(datetime.now() - timedelta(seconds=5))
if sensor.objects.filter(localidad__nombre=loc_name,
fecha__gte=datetime.datetime.now()-timedelta(seconds=2 )).exists():
data.append(sensor.objects.filter(localidad__nombre=loc_name).last().get_m
V())
# print data
else:
data.append(0)
return JsonResponse(data, safe=False)
```

- ***Realiza peticiones al servidor para graficar***

```
function ajax_function(){
var ret=0;
$.ajaxSetup({ cache: false });
$.ajax({
url:"{% url 'flot' %}",
dataType:'json',
data:{'zone':
'{{ loc_obj.get_nombre }}'},
type:'GET',
success:update,
error:function () {
console.log("error");
setTimeout(ajax_function, updateInterval);
}
});
```

Anexo 6 (Uso de librería Flot)

```
div class="row"
div class="col-xs-12"
div class="box box-success"
div class="box-header with-border"
i class="fa fa-bar-chart-o"></i
h3 class="box-title"
div class="box-tools pull-right"
label class="text-muted hidden-xs">Tiempo real</label>
div class="btn-group" id="realtime" data-toggle="btn-toggle"
button type="button" class="btn btn-default btn-xs active" data-
toggle="on">On</button
button type="button" class="btn btn-default btn-xs" data-toggle="off">Off
class="box-body"
div id="interactive" style="height: 300px;"
script
{
var totalPoints = 500;
var updateInterval = 100;
function ajax_function(){
ajaxSetup({ cache: false });
ajax({
url: "{% url 'flot' %}",
dataType: 'json',
data: {'zone':
{{ loc_obj.get_nombre }}'},
type: 'GET',
success: update,
error: function () {
console.log("error");
setTimeout(ajax_function, updateInterval);
}
});
}
var res=[];
function initData() {
for (var i = 0; i < totalPoints; ++i) {
var temp = [i, 0];
res.push(temp)
}
```

Anexo 7 (Configuraciones principales del servidor Apache)

- ***Configuración de servidor Apache***

```
<VirtualHost *:80>
ServerName simora.com.ec
WSGIScriptAlias / /home/juan/PycharmProjects/tesis/tesis/wsgi.py
WSGIDaemonProcess simora.com.ec python-path=/home/juan/PycharmProjects/$
WSGIProcessGroup simora.com.ec
<Directory /home/juan/PycharmProjects/tesis/tesis>
<Files wsgi.py>
Require all granted
</Files>
</Directory
ErrorLog ${APACHE_LOG_DIR}/error.log
CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

- ***Configuración wsgi para la conexión entre apache y la aplicación Django***

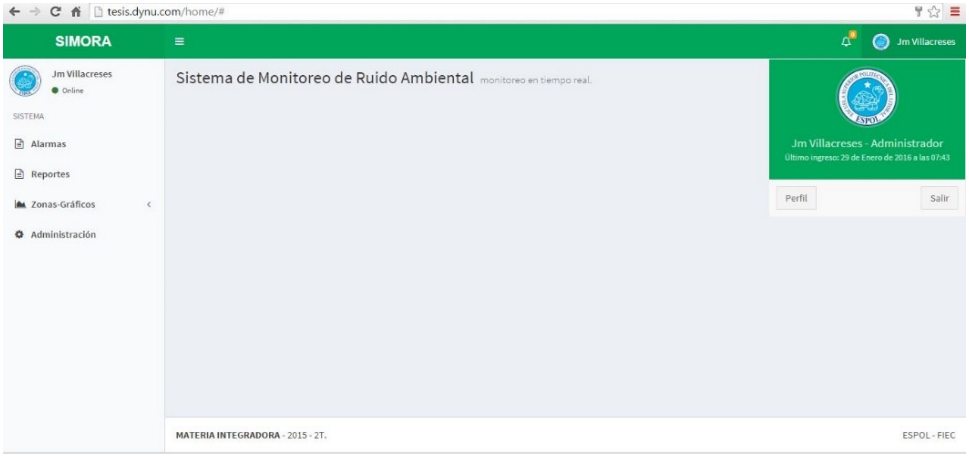
```
import site
import os
import sys
site.addsitedir('/home/juan/.virtualenvs/tesis/lib/python2.7/site-
packages/')
sys.path.append('/home/juan/PycharmProjects/tesis/')
os.environ['DJANGO_SETTINGS_MODULE'] = 'tesis.settings'
from django.core.wsgi import get_wsgi_application
application = get_wsgi_application()
```

Anexo 8 (Presentación de página web SIMORA y validación de usuarios)

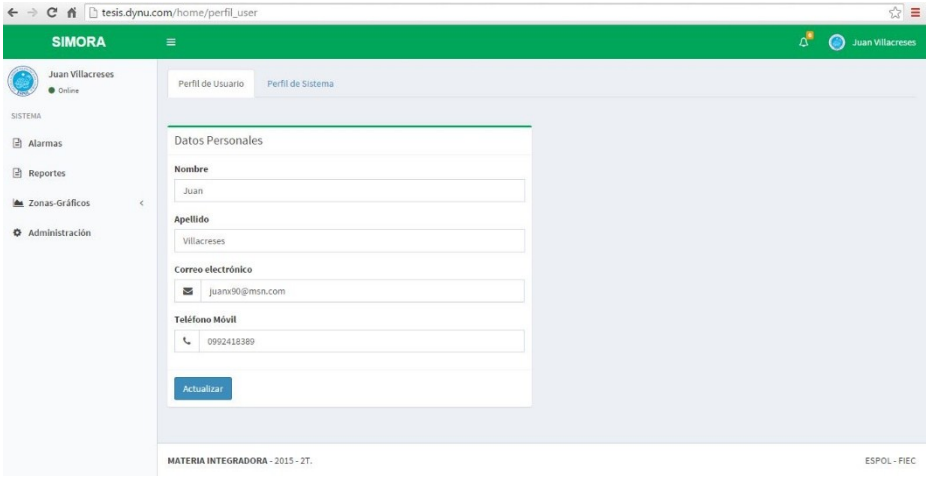
Anexo 9 (Presentación de reestablecer contraseña)

Anexo 10 (Presentación principal de la página web SIMORA)

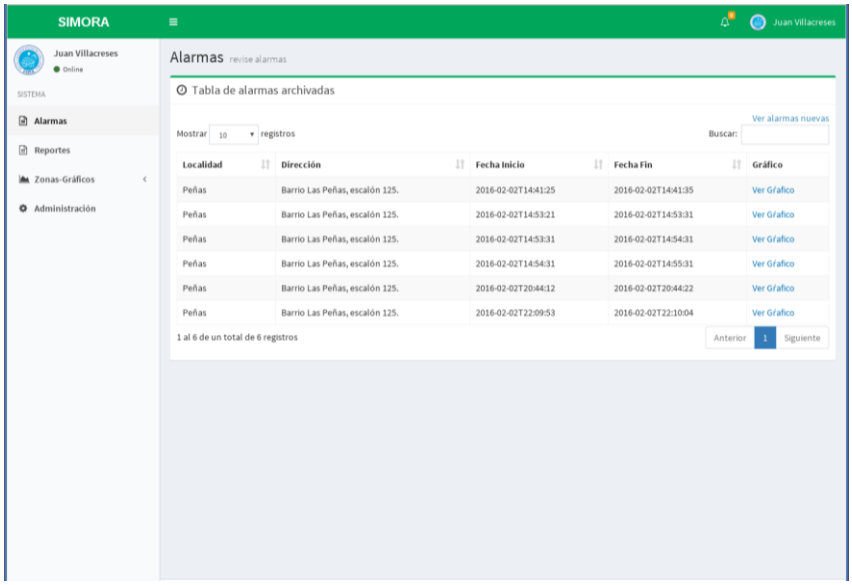
Anexo 11 (Presentación de usuario validado)



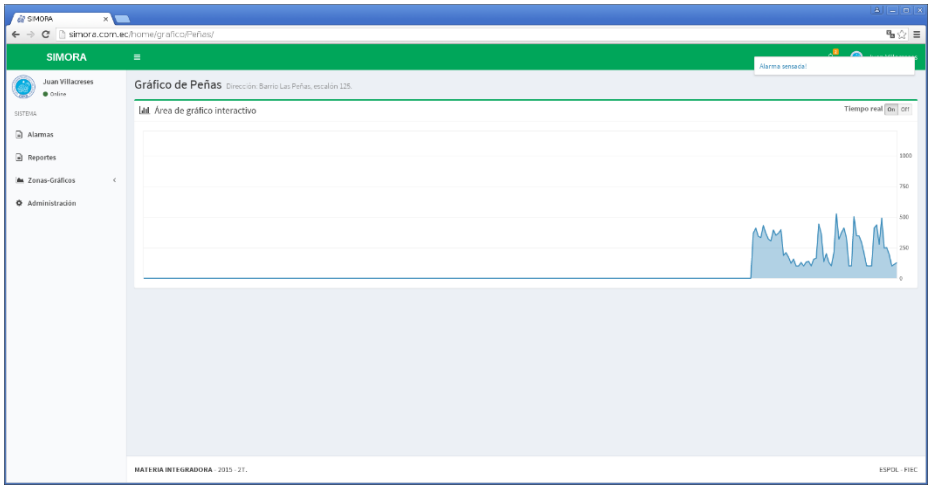
Anexo 12 (Personalizar perfil de usuarios)



Anexo 13 (Alarmas Archivas en la Zona Analizada “Las Peñas”)



Anexo 14 (Grafico a tiempo real generado en “Las Peñas”)



Anexo 15 (Reporte Generado en PDF de “Las Peñas”)