ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

Facultad de Ingeniería en Electricidad y Computación

"SISTEMA ANTIRROBO Y CONTROL DE TEMPERATURA DE EQUIPOS DE CÓMPUTO PARA LOS LABORATORIOS Y CENTRO DE DATOS DE LA FIEC"

INFORME DE MATERIA INTEGRADORA

Previa a la obtención del Título de:

INGENIERO EN TELEMÁTICA

HARLYN STEVEN PICHARDO ORDOÑEZ

IRVING ALEXANDER VALERIANO CANTOS

GUAYAQUIL – ECUADOR

AÑO: 2016

AGRADECIMIENTOS

Primero quiero agradecer a mis padres, Diocles Ariosto Valeriano Veliz y María Elizabeth Cantos Saldarriaga, ya que sin su apoyo incondicional y su gran preocupación por darme una excelente educación no hubiera logrado cumplir esta meta. Les agradezco también por haberme forjado como la persona que soy en la actualidad, este y muchos otros logros se los debo a ustedes que con sus sabias palabras me han ayudado a no rendirme cuando todo parecía estar mal.

También quiero agradecer a mis amigos de la FIEC: Allysson Dávalos, Jorge Ramírez, Steven Pichardo, Malena Anda, Guillermo Zambrano, Jorge Pastor, Daniel Layedra, Alexander García, María Gracia Constante, Joselyne Elizalde, Diana Jácome y Lissette Quimis que fui conociendo a lo largo de la carrera y fueron de gran ayuda mientras se recorría este largo camino compartiendo conocimientos, alegrías y tristezas.

Agradezco también a los tutores de este proyecto Ing. Washington Velásquez e Ing. Marjorie Challen por haberme dado la oportunidad de trabajar con ustedes, ya que sin su apoyo y conocimientos no hubiera podido culminar este proyecto con mi amigo y compañero Harlyn Steven Pichardo Ordoñez.

Por último quiero agradecer a una persona, que aunque no es de la facultad, ha sido un apoyo muy importante durante mi vida universitaria, Katherine Michelle Lomas Ascencio, que con su amistad, compañía y paciencia me ha ayudado a superar varios obstáculos tanto estudiantiles como personales, te agradezco por tanto y espero seguir contando con tu amistad muchos años más.

Irving Valeriano

AGRADECIMIENTOS

Mis más sinceros agradecimientos a Dios por darme salud y conocimientos en todos estos años, a mis padres por su gran esfuerzo y su apoyo sin ellos capaz no hubiese llegado tan lejos como ahora, a mi familia por sus ánimos y su cariño, a mis mejores amigos especialmente a mi mejor amiga de la universidad Malena por permitirme sonreír día a día y su apoyo incondicional.

Agradezco a mis profesores de la carrera que me dieron la inspiración para poder seguir adelante y no detenerme en el camino tal que pueda lograr mis metas académicas con éxito.

Agradezco también a mis compañeros del DST en especial a los asistentes que nos brindaron su apoyo en el proyecto.

Harlyn Pichardo

DEDICATORIA

El presente proyecto lo dedico a mis padres, Diocles Ariosto Valeriano Veliz, María Elizabeth Cantos Saldarriaga y a mi hermana Ingrid Antonella Valeriano Cantos que son el motor de inspiración en mi vida por su gran compañía y su inmenso amor hacia mí. Ustedes son mi más grande alegría.

Irving Valeriano

El presente proyecto lo dedico a Dios, a mis padres, a mi abuelita Emma que desde el cielo me protege y sonríe por mí, a mi familia y a mis mejores amigos sin ellos nada de esto sería posible.

Harlyn Pichardo

TRIBUNAL DE EVALUACIÓN

Ing. Washington Velásquez, M.Sc.

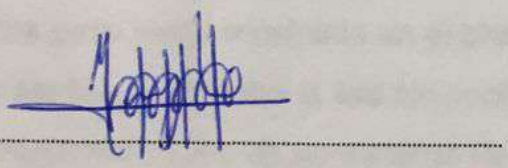
PROFESOR EVALUADOR

Ing. Marjorie Chalén, MSIG.

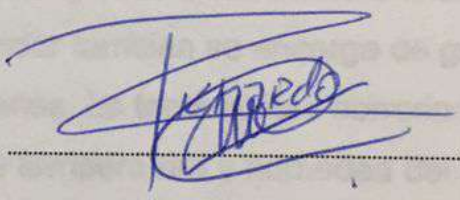
PROFESOR EVALUADOR

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"



Irving Alexander Valeriano Cantos



Harlyn Steven Pichardo Ordoñez

RESUMEN

En este documento se describe una solución para el monitoreo de temperaturas en los equipos y el centro de datos de la FIEC para prevenir daños por sobrecalentamiento, así mismo, se controla que los periféricos no sean desconectados por agentes externos. Esta implementación se divide en 3 fases descritas a continuación:

La primera fase corresponde a una aplicación web adaptiva que permite administrar la información del sistema y visualizarlo desde cualquier ordenador. La segunda fase consiste en la creación de una infraestructura servidor-cliente, tal que, el cliente envía la información de cada estación de trabajo al servidor y esta lo almacena a la base de datos para luego mostrarla en el sitio web; el servidor también se encarga de generar las alertas y enviarlas a sus respectivos destinatarios. La tercera fase corresponde a la implementación de un sistema de monitoreo de temperatura y humedad del centro de datos utilizando *Arduino* para que cense datos y los envíe a almacenar para realizar los mensajes correspondientes si existiese algún problema en el centro de datos de la facultad.

Para realizar este proyecto se utilizaron varios lenguajes de programación tales como: php, C#, javascript, html, entre otros, por lo que para comunicarse entre sí se utilizó websockets que consiste en una comunicación mediante paquetes de información en la red utilizando el protocolo *TCP*. Para generar las alertas de *SMS* se utiliza un Api llamado Textmagic que al recargarlo con dinero este nos permite enviar mensajes de texto a cualquier dispositivo móvil de los diferentes países del mundo.

GLOSARIO DE TERMINOS

HTML:	HyperText Markup Language, lenguaje de marcado para la elaboración de sitios web.
PHP:	Lenguaje de programación orientado al diseño web.
DBMS:	Sistema de gestión de base de datos.
WEBSOCKETS:	Proporciona comunicación full-duplex utilizando protocolo TCP/UDP.
TCP:	Protocolo de control de transmisión de datos.
SMS:	Servicio de mensajes cortos que son usados para comunicación entre dispositivos móviles.
API:	Interfaz de programación de aplicaciones.
PS/2:	Conector de datos de teclado y monitor.
USB:	Bus universal de serie, sirve para comunicar y proveer energía a dispositivos electrónicos.
PSQL:	Postgres SQL sistema de gestión de base de datos.
FRAMEWORK:	Infraestructura digital, estructura conceptual definido, contiene módulos concretos de software.
IP:	Protocolo de internet, dirección lógica de un dispositivo que se encuentre en una red definida compuesta por 4 octetos comprendidos entre 0-255 cada uno.

ÍNDICE GENERAL

CAPÍTULO 1

1. PLANTEAMIENTO DEL PROBLEMA.

La Facultad de Ingeniería en Electricidad y Computación (FIEC) cuenta con varios laboratorios de computación, creados con la finalidad de ofrecer un soporte en el desarrollo de las actividades académicas de profesores y estudiantes mediante el préstamo de computadores para el dictado de clases o seminarios.

La frecuencia con la que los usuarios utilizan este servicio es considerable y resulta dificultoso para los ayudantes de turno, asistentes y personal del departamento de soporte técnico vigilar constantemente los periféricos de las diferentes estaciones de trabajo. Se ha dado el caso en donde los periféricos se sustraen de los equipos generando molestias a los usuarios que les dan un buen uso y al respectivo encargado de laboratorio en la realización del informe final de los inventarios de cada laboratorio.

En el caso de que un procesador de un equipo este trabajando en condiciones climáticas no apropiadas puede dañarse, causando un problema mayor ya que ésta parte del equipo es aún más costosa.

Para que los equipos de la facultad puedan estar comunicados entre sí y tengan acceso a internet el Departamento de Soporte Técnico (DST) de la FIEC, cuenta con una sala de rack en donde se encuentran los servidores y equipos de conmutación y enrutamiento necesarios. Al igual que los computadores normales, los equipos en la sala de rack también deben operar a una temperatura apropiada.

El monitoreo constante de los equipos sin una herramienta administrativa para este tipo de problemas, hace que el trabajo se dificulte y en mayor detalle en la sala de rack, lo cual se vuelve un poco engorroso el proceso de administración de todas las herramientas tecnológicas que están a cargo del DST en la facultad.

1.1 Descripción del problema

Se necesita de un sistema que monitoree la temperatura en los distintos laboratorios de computación y así mismo en la sala de RACK con la diferencia que en esta sala se tendrá que controlar la humedad del cuarto, para que cuando

existan fallas de acondicionamiento se pueda alertar al administrador y se eviten daños en los equipos por sobrecalentamiento.

Adicional, se debe implementar un cliente que permita controlar la temperatura de las computadoras y evitar el robo de equipos de computación, para esto se debe avisar al administrador en qué momento se desconecta una parte del computador mediante alertas vía correo he incluso a través de un módulo GSM.

Para llegar a la implementación debe tener en cuenta lo siguiente:

- Se debe constar con un módulo que permita conocer la temperatura en el los laboratorios de computación.
- Se debe tener un módulo que permita conocer la temperatura, la humedad en la sala de RACK.
- Implementar un módulo de alarmas y GSM para enviar al administrador en caso de alguna incidencia.
- El sistema debe constar con un cliente (software) que controle el robo de algún dispositivo y la temperatura del computador.
- El sistema debe constar con un servidor que este monitoreando las computadoras y la sala de rack.
- Se debe implementar una aplicación web adaptativa para que el administrador pueda observar con facilidad la temperatura del computador y la información en caso de algún hurto.

1.2 Justificación del problema

Debido a los robos ocurridos en los laboratorios de computación de la facultad, ya sea por los mismos estudiantes o por personas ajenas al laboratorio, y dado que no se puede monitorear todo el tiempo los laboratorios, este proyecto le dará solución a ese problema mediante un sistema de monitoreo, tanto de temperatura como de control de periféricos del ordenador, el cual mediante una interfaz web se muestra la información de los ordenadores, de la temperatura y humedad de la sala de servidores de la Facultad de Ingeniería en Electricidad y Computación.

Para garantizar el buen funcionamiento de los equipos, éstos deben tener sus respectivos periféricos conectados y estar a una temperatura adecuada. Además para que estos equipos puedan estar siempre conectados, los dispositivos de comunicación deben estar operativos a cada momento.

Así, cuando el sistema detecte que ha ocurrido un problema en alguno de los equipos de los laboratorios o de la sala de rack, éste notifique de lo ocurrido al o los responsables de llevar el control de los mismos.

1.3 Objetivos

1.3.1 Objetivo general

Implementar un sistema antirrobo que cuente con monitoreo de conexión de dispositivos de los equipos de cómputo de los laboratorios. Adicionalmente, permite conocer la temperatura y humedad del centro de datos de la facultad mediante un sensor, generando alertas en el caso de que existan algunos inconvenientes con la misma, utilizando hardware y software libre que permita reducir gastos innecesarios en la adquisición de equipos.

1.3.2 Objetivos específicos.

- Desarrollar un sistema cliente-servidor para el monitoreo de los equipos de los laboratorios de computación.
- Implementar una interfaz para la administración del sistema en el cual, puedan ver la información de los equipos que se encuentran monitoreando.
- Implementar un módulo electrónico que cense la temperatura y humedad de la sala de rack para su posterior almacenamiento en una base de datos.
- Implementar un módulo que permita realizar alertas vía SMS para notificar problemas.

1.4 Limitaciones y alcance del sistema

- Dado que los periféricos PS/2 al desconectarse del registro no actualiza de forma inmediata su estado, el monitoreo para estos tipos de ordenadores se los realiza al inicio de Windows ya que si existe un cambio de estos que se registra al iniciar el sistema operativo, en cambio en dispositivos USB lo realiza de forma inmediata.
- Las alertas se generan con un retraso, en el caso de correo electrónico debe ser procesado por el servidor de la facultad para poder realizar él envió adecuado.
- Las alertas de SMS se envían de inmediato, pero éste utiliza un número internacional por lo que puede haber cobro de recibo del mensaje.
- El sistema está orientado para que sea instalado en todos los laboratorios que operan dentro de la facultad, tanto para uso de alumnos en general como para uso de clases, por lo que debe estar en el mismo segmento de red de ESPOL para que exista una comunicación exitosa.

CAPÍTULO 2

2. MARCO TEÓRICO.

2.1 HTML (Hyper Text Markup Language)

Es un lenguaje en el cual se desarrollan las páginas web. Es un estándar de hipertexto, esto quiere decir que permite escribir texto de forma estructurada para la definición de contenido de una página web como texto, imágenes, videos entre otros. Un documento HTML no solo se compone de texto, sino también de imágenes, sonido, videos, y demás, como resultado puede obtenerse un documento de tipo multimedia.

HTML se escribe en forma de etiquetas rodeadas de corchetes tal que también puede describir la apariencia y flujo de una página web. Además puede tener referencia a un tipo de programa llamado script y que puede afectar el comportamiento de los browsers y otros intérpretes de código HTML. [1]

2.2 CSS (Cascading Style Sheets)

Es un lenguaje para definir y crear la presentación de un documento estructurado escrito en HTML, es el encargado de elaborar las especificaciones de hojas de estilo de un sitio web o un documento en específico, en otras palabras CSS es el encargado de darle formato a los sitios HTML tales como estilos, colores, márgenes, estilo de párrafos, entre otros.

La idea central del desarrollo de CSS es separar la codificación de estructura de un documento de su presentación del mismo ya que así se facilita la elaboración de sitios web y su edición. [2]

2.3 JAVASCRIPT

Es un lenguaje de programación interpretado por lo que no se necesita compilar los programas para poder ejecutarlos, se utiliza principalmente para crear páginas web dinámicas.

Se define como orientado a objetos, basado en prototipos, imperativo, débilmente tipado y dinámico, utilizado frecuentemente en el lado del cliente

implementado como parte del navegador permitiendo una mejor interacción con las páginas web que hacen uso de esta herramienta. [3]

2.4 BOOTSTRAP

Es el más popular HTML, CSS y JavaScript framework para desarrollo responsive y primeros proyectos móviles en la web. Bootstrap hace que el desarrollo front-end web sea más rápido y más fácil. Está hecho para usuarios de todos los niveles en cuanto a conocimientos de estos lenguajes y para dispositivos de todas las formas y proyectos de todos los tamaños. [4]

2.5 PHP

Es un lenguaje de programación en el que se pueden realizar todo tipo de cosas tales como evaluar datos de un formulario, validar páginas de login, consultas a bases de datos, recibir cookies o pequeños paquetes de datos que tu navegador usa para recordar cosa, como por ejemplo si te registraste en una página especifica.

PHP se considera como uno de los lenguajes de programación más flexibles, potentes y de alto rendimiento conocido, por lo que es usado por la mayoría de programadores para poder desarrollar sus aplicaciones web, puede ser desplegado en la mayoría de servidores web y en la mayoría de los sistemas operativos y plataformas sin ningún costo. [5]

2.6 VISUAL C#

Es un lenguaje de programación orientado a objetos desarrollado y estandarizado por Microsoft, diseñado para compilar diversas aplicaciones que se ejecutan en .NET Framework. La biblioteca de clases de .NET Framework ofrece acceso a numerosos servicios de sistema operativo y a otras clases útiles y adecuadamente diseñadas que aceleran el ciclo de desarrollo de manera significativa. C# es eficaz, con seguridad de tipos y orientado a objetos. C# está diseñado para la infraestructura de una interpretación común por lo cual se lo considera un lenguaje de fácil aprendizaje. [6]

2.7 POSTGRESQL

Es un potente sistema de base de datos objeto-relacional de código abierto. Incluye una biblioteca de funciones estándar con cientos de funciones integradas que van desde las operaciones matemáticas básicas, operaciones con strings para criptografía y compatibilidad con Oracle. Los disparadores (triggers) y procedimientos almacenados pueden ser escritos en C y se cargan en la base de datos como una biblioteca, lo cual permite una gran flexibilidad y ampliación de sus capacidades. [7]

2.8 ARDUINO

Es una plataforma de prototipo electrónica de código abierto (open-source) basada en hardware y software flexibles y fáciles de usar. Está pensado para artistas, diseñadores, como hobby o a su vez para crear objetos o entornos interactivos, con la ayuda de otros elementos electrónicos como sensores ya sean de temperatura, humedad, entre otros, puede medir el entorno que lo rodea.

El hardware consiste en una placa con un microcontrolador *Atmel AVR* y puertos de entrada/salida. Los microcontroladores más usados son el Atmega168, Atmega328, Atmega1280, y Atmega8 por su sencillez y bajo coste que permiten el desarrollo de múltiples diseños. Por otro lado el software consiste en un entorno de desarrollo que implementa el lenguaje de programación *Processing/Wiring* y el cargador de arranque que es ejecutado en la placa. El código escrito y compilado en el ordenador es grabado en el de la placa la cual con ayuda de elementos electrónicos monitorea variables del medio y controla otros dispositivos. [8]

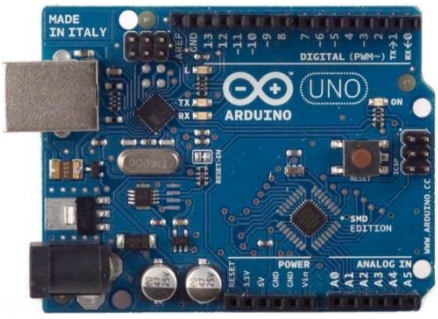


Figura 2.1: Placa Arduino UNO.

2.9 MÓDULO DHT11

El DHT11 es un sensor básico digital de medición de temperatura y humedad. Este sensor está basado en un termistor que sirve para medir el aire circundante (temperatura) e implementa un sensor interno capacitivo para la medición de humedad. Su implementación es bastante sencilla, pero se requiere una cuidadosa sincronización para la toma de datos, es usado para aplicaciones en las que necesita medir los niveles de temperatura y humedad de ciertos ambientes. [9]

Características:

- Tensión de alimentación: +5 V
- Rango de temperatura: 0-50 ° C error de ± 2 ° C
- 20-90% RH Humedad ± 5% RH error.
- Interfaz: Digital.

Figura 2.2: Módulo DHT11

2.10 SHIELD ETHERNET ARDUINO

Es un módulo que permite la conexión a internet de una placa de *Arduino* que permite la interacción del dispositivo en la red como si fuera un ordenador más.

Su conexión es fácil, utilizando un cable RJ45 y siguiendo las instrucciones sencillas con comandos específicos en lenguaje C para poder configurarle una dirección IP, dirección MAC, una máscara de red, puerta de enlace, DNS se logra que la placa se agregue a la red. Sus características son: [10]

- Requiere placa *Arduino*
- Voltaje de operación: 5v

- Controlador Ethernet: W5100 con un buffer interno de 16k
- Velocidad de conexión: 10/100 Mbps
- Conexión con *Arduino* mediante puerto SPI.

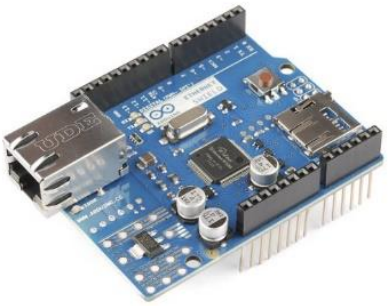


Figura 2.3: Ethernet Shield Arduino.

2.11 TEXTMAGIC

Es un servicio de mensajería de texto de negocios para el envío de notificaciones, alertas, recordatorios, confirmaciones y las campañas de marketing SMS, consta las siguientes características:

- Envía mensajes de texto en línea a más de 200 países alrededor del mundo.
- Convierte email a SMS y envía a cualquier dispositivo móvil.
- Integra un api SMS para empresas para crear una aplicación propia de sistema de alertas. [11]

CAPÍTULO 3

3 IMPLEMENTACIÓN Y DESARROLLO.

La solución al problema descrito se divide en tres partes; La primera consiste en una aplicación web, donde el administrador de los laboratorios podrá monitorearlos desde su ordenador de trabajo o cualquier otro computador de la facultad, los equipos que se deseen inspeccionar deben tener instalado el servicio creado para este fin.

La segunda parte se basa en la implementación de una infraestructura servidor-cliente que es la encargada del intercambio de información entre los computadores de los laboratorios y el servidor para que ésta pueda ser almacenada y luego consultada.

El cliente está diseñado de tal forma que cada cierto tiempo obtenga información de los periféricos y la temperatura que serán enviados al servidor. El servidor está formado por dos scripts, uno se encarga de recibir la información que se encuentra enviando los computadores de los laboratorios y además se encarga de analizarla para la generación de alertas, el otro se encuentra monitoreando por si se generan alertas y se encarga de enviarlas.

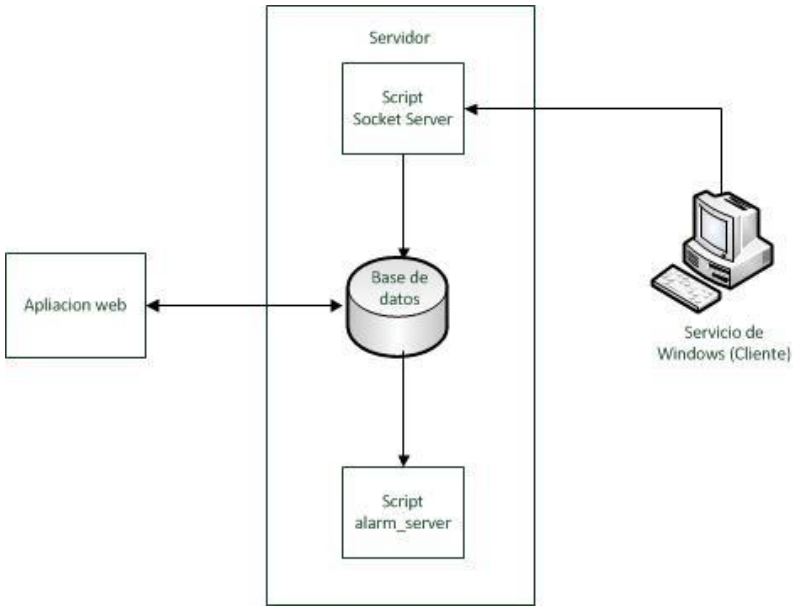


Figura 3.1: Diagrama de bloques fase uno y dos.

La tercera parte consiste en la operación de un sensor de temperatura y humedad, el cual toma datos del cuarto de servidores y los almacena en la base datos para su monitoreo en la aplicación web. Si la temperatura o la humedad se encuentran en un nivel fuera del rango permitido, el sistema genera una alerta y se notifica mediante correo electrónico y SMS.

Figura 3.2: Diagrama de bloques fase tres.

3.1 Aplicación Web Adaptiva

El diseño de la aplicación web se lo realizó mediante la programación con los lenguajes HTML, PHP y JAVASCRIPT utilizando un framework llamado *Bootstrap* para crear de una manera más sencilla un sitio web. Para la manipulación de los datos de la base desde la interfaz web se utilizó PHP, además para crear gráficos ciertas validaciones se utilizó javascript.

3.1.1 Modelo de base de datos

El gestor de bases de datos escogido es *postgresql* ya que permite utilizar multiprocesos, garantizando la estabilidad del sistema. En la figura 3.3 se observa el modelo utilizado.

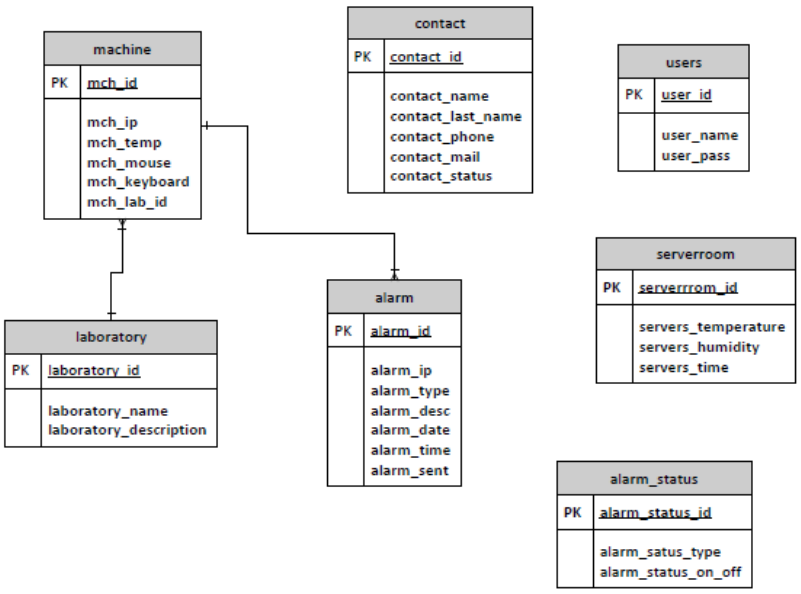


Figura 3.3: Modelo Entidad-Relación.

La figura 3.3 muestra el diseño de la base de datos que utiliza el sistema el cual, consta de siete tablas tal que cada una almacena datos específicos según se requiere, por lo que a continuación se explica para que sirven cada una de estas tablas creadas:

La tabla ***machine*** almacena todos los datos que se requiere de un computador tales como: laboratorio al que pertenece, dirección IP, temperatura, estado del mouse y teclado que pueden ser conectado o desconectado.

La tabla ***laboratory*** guarda el nombre del laboratorio y dar una breve descripción sobre el funcionamiento del mismo.

La tabla ***contact*** registra la información de las personas a las cuales se les enviará las alertas las cuales contienen: nombre, apellido, correo electrónico, número de teléfono y un campus status para saber si dicho contacto está habilitado o no para recibir las alertas.

La tabla ***alarm*** almacena las alertas generadas por el script del servidor con los siguientes datos: dirección IP del ordenador origen, tipo de alarma,

descripción, fecha de generación, hora de generación, y un campo que indica si la alarma ya fue enviada o no a los destinatarios.

La tabla ***users*** guarda los usernames de los usuarios con sus respectivas contraseñas para el ingreso al sistema.

La tabla ***serverroom*** registra la temperatura y humedad de la sala de rack y su respectiva fecha de censado.

La tabla ***alarm_status*** almacena el tipo de notificaciones que se generan vía correo o SMS.

3.1.2 Ingreso al sistema

Como se muestra en la figura 3.4 la ventana de autenticación para los administradores del sistema. Solo los usuarios almacenados en la tabla *users* están autorizados a agregar, eliminar o modificar datos, así como de las demás características que ofrece el sistema.

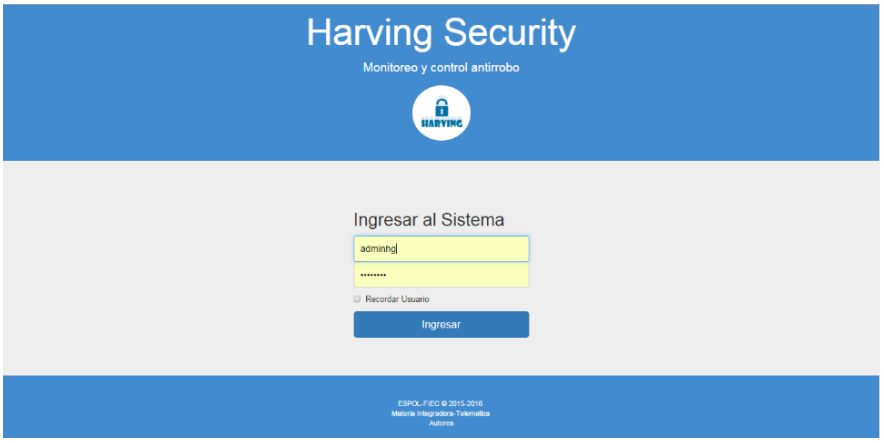


Figura 3.4: Pantalla de ingreso al sistema Harving.

3.1.3 Conexión a la base de datos

Para la conexión a la base de datos se utilizó funciones de PHP exclusivas para *postgresql*, tal que se le indica el nombre de base de datos, usuario y contraseña del servidor para poder acceder al mismo.

```
$conexion = pg_connect("host=200.9.176.13 dbname=harvingdb
user=harving password=admin123") or die('No se ha podido conectar:
' . pg_last_error());
```

La conexión se realiza en un archivo *conexion.php* dentro del servidor web el cual es llamado en todas las páginas en las que se realicen consultas a la base de datos.

Como se puede observar se utilizó la función pg_connect que permite realizar una conexión al servidor mediante su dirección IP y así acceder a los datos del sistema.

3.1.4 Escoger Laboratorio

Esta ventana fue implementada para mostrar de forma gráfica los laboratorios de computación ingresados al sistema con sus respectivos ordenadores en el que se muestra dirección IP, temperatura, mouse, teclado y si el ordenador esta encendido o apagado.

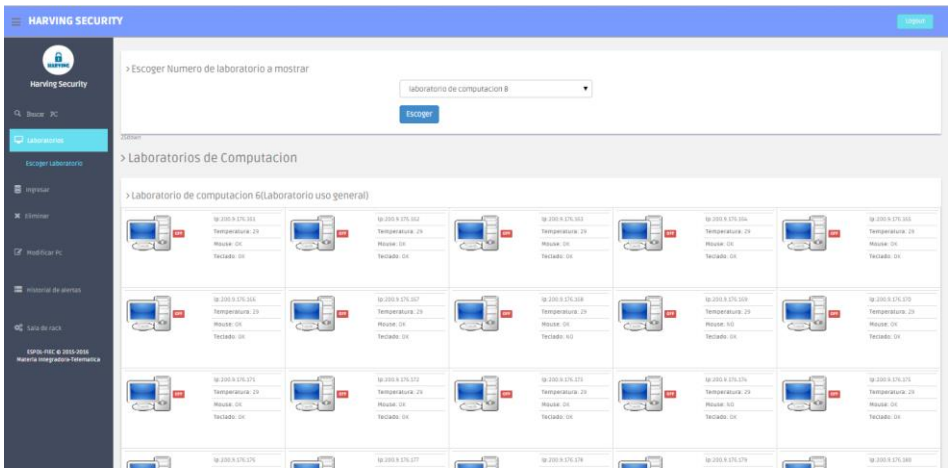


Figura 3.5: Pantalla Escoger Laboratorio.

Como se puede observar en la figura 3.5, el usuario debe escoger el nombre de laboratorio que desee mostrar, para hacer posible esto se utilizó un query en el que se realiza una búsqueda de todos los ordenadores que pertenecen a ese laboratorio:

```
$sentenciaSQL = "SELECT * FROM machine where mch_lab_id='$lab'
order by mch_ip asc";
```

Para el envío de datos de las páginas se utilizó el método POST que usa HTML para el ingreso de datos mediante formularios web. Este método permite que los datos se envíen de manera interna y así no aparezcan los datos enviados dentro del URL de la página solicitada.

Para saber si un ordenador se encuentra encendido o apagado, se utilizó una función en PHP que le permite hacer ping a una dirección IP, si el ping no responde, entonces se asume que el ordenador se encuentra apagado, caso contrario; si el ping devuelve un tiempo de retorno, entonces se plantea que el ordenador se encuentra encendido y está dentro de la red.

La función ping se implementó de la siguiente manera:

```
function ping($host, $port, $timeout)
{
  $tB = microtime(true);
  $fP = fSockOpen($host, $port, $errno, $errstr, $timeout);
  if (!$fP) { return "down"; }
  $tA = microtime(true);
  return round((($tA - $tB) * 1000), 0)." ms";
}
```

La función ping realiza una prueba de conexión a un ordenador en específico, si este no responde, esta retorna un valor de down caso contrario, responde con un tiempo en milisegundos de la conexión, muy parecido al comando ping de windows.

3.1.5 Ingresar laboratorio

Esta ventana fue implementada para que el usuario pueda ingresar un nuevo laboratorio al sistema mediante un sencillo formulario HTML.

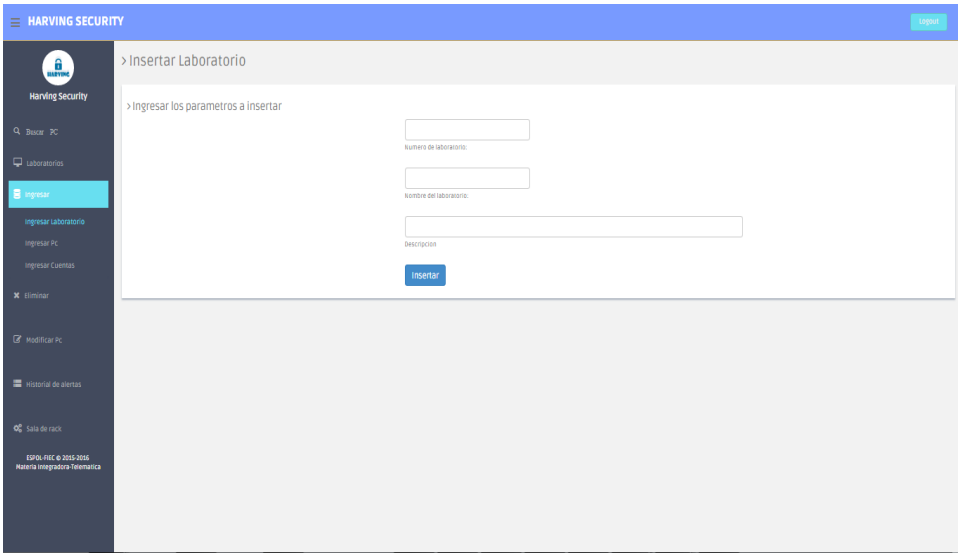


Figura 3.6: Pantalla ingresar Laboratorio.

Como se puede observar en la figura 3.6, el usuario ingresa el número de laboratorio, el nombre y una breve descripción de su funcionamiento.

Para ingresar los datos a la base de datos se utilizó el siguiente query:

```
$sentenciaSQL = "INSERT INTO laboratory
(laboratory_name,laboratory_id,laboratory_description) VALUES
('$lab','$nlab','$description')";
```

Las variables $lab, $nlab y $description corresponden a los datos enviados del formulario obtenidos mediante el método POST al pulsar el botón enviar de la ventana.

3.1.6 Ingresar Pc

Esta ventana fue creada para que el usuario pueda ingresar un nuevo ordenador a un laboratorio en específico mediante un formulario.

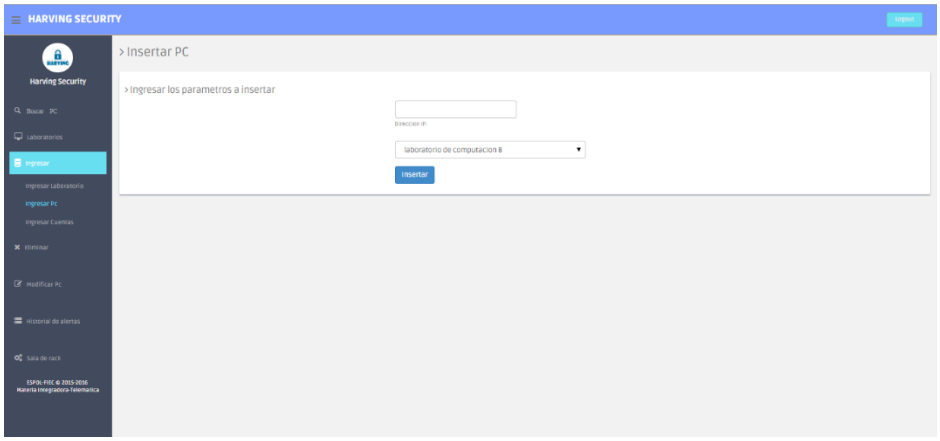


Figura 3.7: Pantalla ingresar Pc.

Como se puede observar en la figura 3.7, el usuario ingresa la dirección ip del nuevo ordenador y escoge el laboratorio al que pertenece.

Para ingresar los datos a la base de datos se utilizó el siguiente query:

```
$sentenciaSQL = "INSERT INTO machine (mch_ip,mch_lab_id) VALUES
('$ip','$lab')";
```

Las variables $ip y $lab corresponden a la dirección IP y el laboratorio al que pertenece dicho ordenador el cual son obtenidos posteriormente al envío del formulario al Sistema.

3.1.7 Eliminar Laboratorio y Eliminar Pc

Estas ventanas permiten que el usuario pueda eliminar un laboratorio mediante su nombre y también un ordenador que haya ingresado de manera incorrecta al sistema.

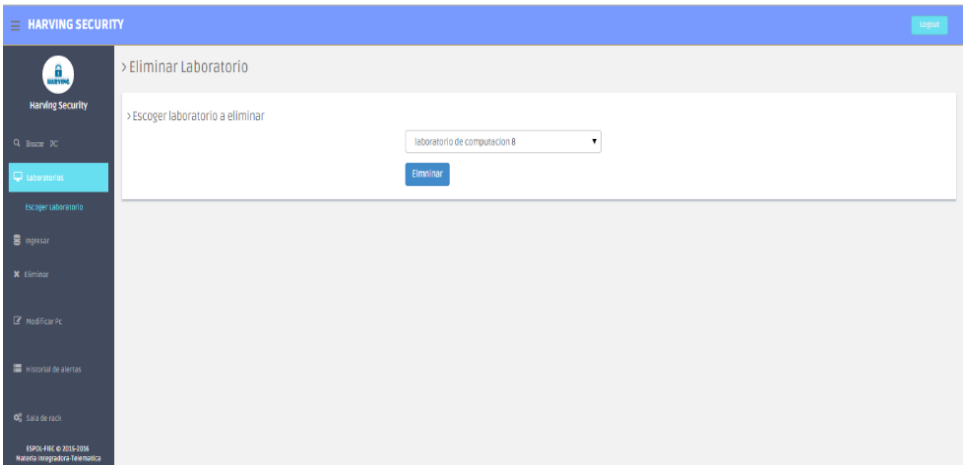


Figura 3.8: Pantalla eliminar laboratorio

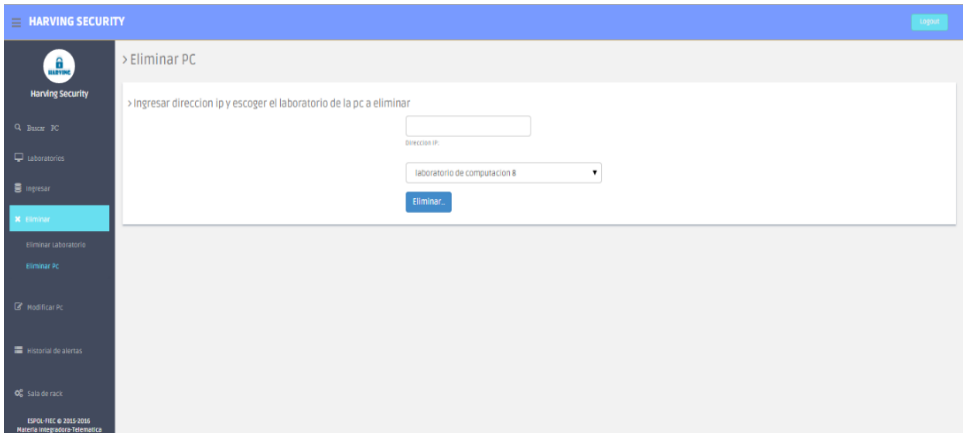


Figura 3.9: Pantalla eliminar Pc.

Como se puede observar en la figura 3.8, el usuario debe escoger un laboratorio para poder ser eliminado, para realizar esta operación se ejecuta el siguiente query:

```
$sentenciaSQL = "DELETE FROM laboratory WHERE laboratory_id =
'$nlab'";
```

Como se puede observar en la figura 3.9, el usuario ingresa la dirección IP del ordenador y también tiene que escoger el laboratorio al que pertenece para poder ser eliminado del sistema, esta operación se la realizó con el siguiente query:

```
$sentenciaSQL = "DELETE FROM machine  WHERE mch_ip = '$ip' AND
mch_lab_id = '$lab'";
```

3.1.8 Modificar Pc

Esta ventana permite al usuario modificar un ordenador que está ingresado en el sistema con el fin de corregir algún error que haya cometido al ingresar un ordenador a un laboratorio.

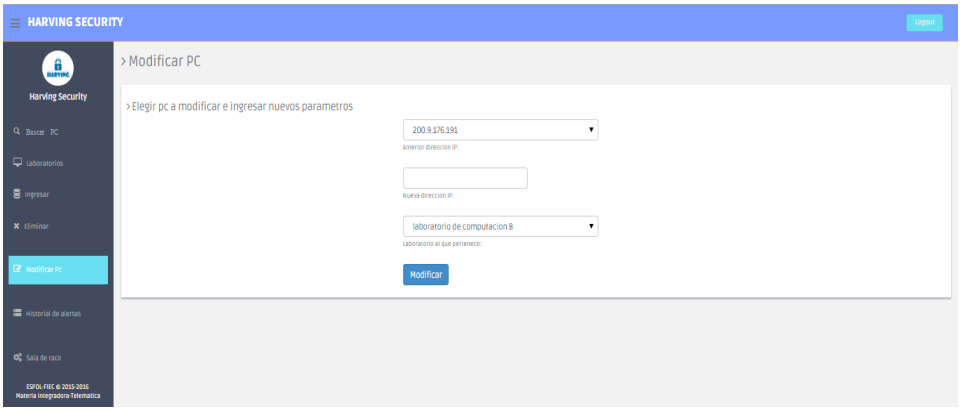


Figura 3.10: Pantalla modificar PC

La figura 3.10 presenta el ingreso por parte del usuario para que seleccione la dirección IP del ordenador que desea modificar e ingresar los nuevos valores.

La modificación se la realizó mediante el siguiente query:

```
$sentenciaSQL = "UPDATE machine SET mch_ip='$ip2',
mch_lab_id='$lab' WHERE mch_ip='$ip'";
```

3.1.9 Historial de Alertas

Esta ventana permite mostrar un historial de alertas generadas en un intervalo de tiempo ingresado por el usuario. El historial consiste en mostrar un resumen ejecutivo donde indica la persona que está a cargo del laboratorio y el intervalo de tiempo del mismo, además presenta una gráfica lineal donde indica cuantas alertas ha tenido un ordenador.

Para que el usuario pueda escoger una fecha se utilizó una herramienta en JAVASCRIPT llamada *Datepicker,* el cual se lo utilizó de la siguiente manera:

```
<script>
  $(document).ready(function() {
    $("#datepicker").datepicker({
    dateFormat: 'yy-mm-dd',
    startDate: '-3d'
   });
  });
  </script>
```

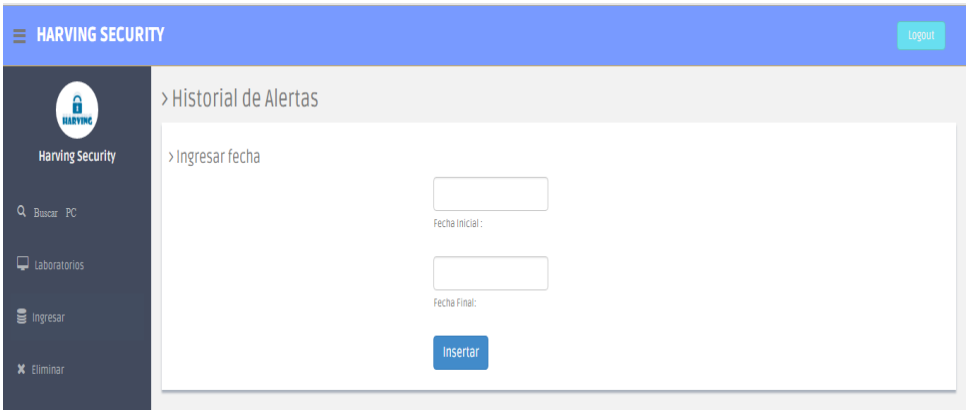


Figura 3.11: Pantalla historial de alertas

La figura 3.11 muestra la interfaz donde el usuario configura el rango de fechas para poder observar los reportes.

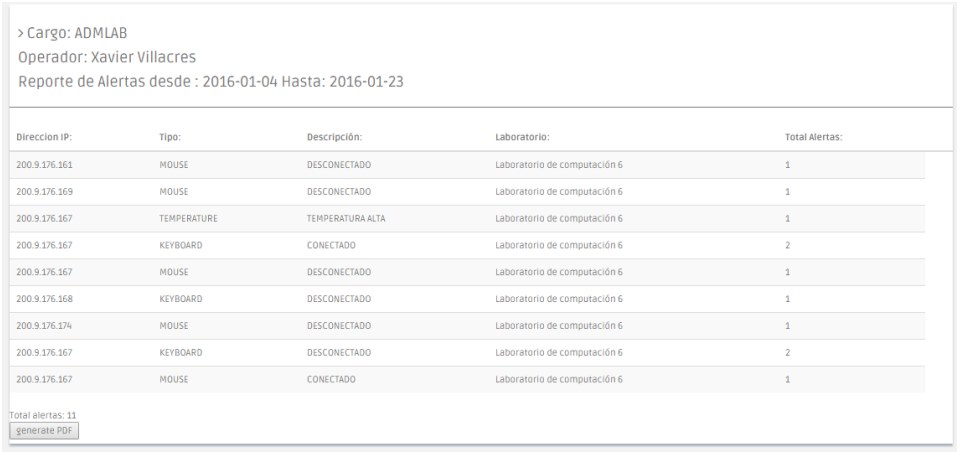


Figura 3.12: Historial de alertas resumen ejecutivo

La figura 3.12 muestra el resumen ejecutivo mediante una tabla que indica la dirección ip, tipo, descripción y el número de alertas en ese intervalo de un ordenador en específico.

Cargo: ADMLAB

Operador: Xavier Villacres

Reporte de Alertas desde : 2016-01-04 Hasta: 2016-01-23

Direccion IP:	Tipo:	Descripción:	Laboratorio:	Total Alertas:
200.9.176.169	MOUSE	DESCONECTADO	Laboratorio de computación 6	1
200.9.176.167	TEMPERATURE	TEMPERATURA ALTA	Laboratorio de computación 6	1
200.9.176.167	KEYBOARD	CONECTADO	Laboratorio de computación 6	2
200.9.176.167	MOUSE	DESCONECTADO	Laboratorio de computación 6	1
200.9.176.168	KEYBOARD	DESCONECTADO	Laboratorio de computación 6	1
200.9.176.174	MOUSE	DESCONECTADO	Laboratorio de computación 6	1
200.9.176.167	KEYBOARD	DESCONECTADO	Laboratorio de computación 6	2
200.9.176.167	MOUSE	CONECTADO	Laboratorio de computación 6	1

Total alertas: 11

Figura 3.13: Historial de alertas archivo pdf

La figura 3.13 muestra el formato pdf del resumen ejecutivo para poder imprimir y presentarlo al jefe del laboratorio.

3.1.10 Sala de Rack

Esta ventana fue implementada para mostrar un gráfico lineal de la temperatura y humedad de la sala de servidores comprendido en la última hora.

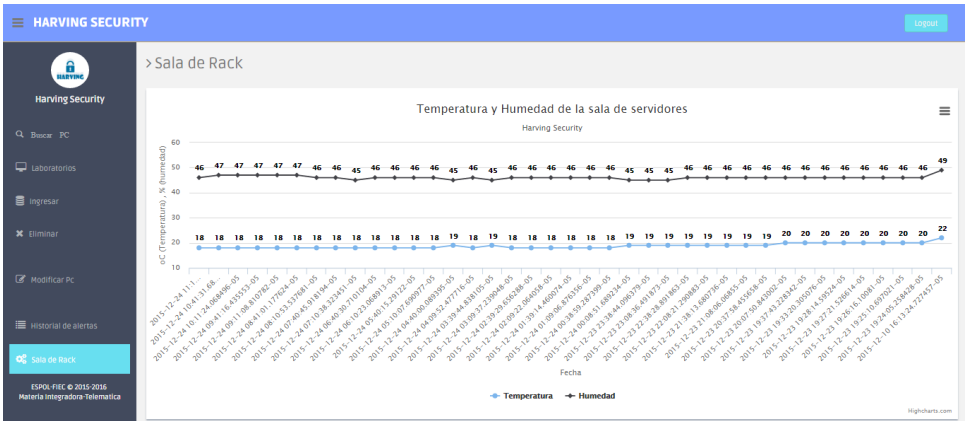


Figura 3.14: Pantalla Sala de Rack.

La figura 3.14 presenta la gráfica de temperatura y humedad de la sala de servidores, para implementar una gráfica en html se utilizó la librería charts de jquery interna de javascript, lo fundamental de dicho script es la configuración los datos para el eje x y el eje y el cual se muestra a continuación:

```
xAxis: {
                        categories: [<?php for
($i=0;$i<pg_num_rows($resultado);$i++){ echo "'$dato3[$i]',";
}?>],
                        // Pongo el título para el eje de las 'X'
                        title: {
                               text: 'Fecha'
                        }
                }
```

Los datos se los obtuvo al ejecutar el siguiente query:

```
$sentenciaSQL = "SELECT * FROM serverroom order by servers_time
desc LIMIT 60  ";
```

3.1.11 Ingresar Cuentas

Esta ventana fue implementada para que el usuario ingrese al sistema nuevas cuentas de contacto los cuales reciben las alertas generadas por el servidor.

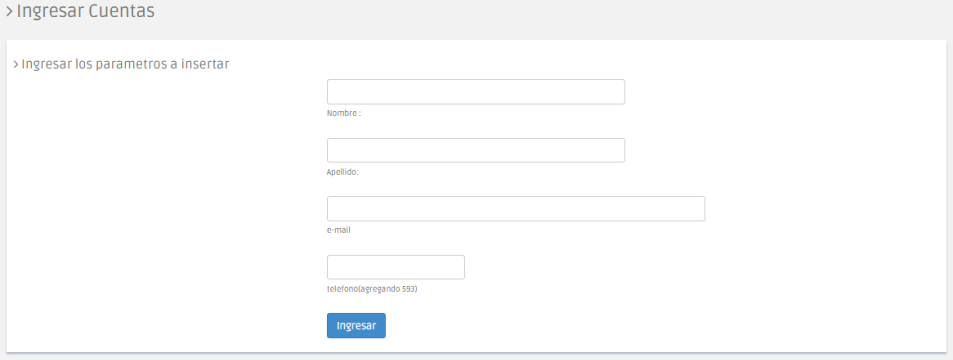


Figura 3.15: Pantalla Ingresar cuentas

La figura 3.15 presenta el modo en el que el usuario debe ingresar los datos correspondientes a un nuevo contacto tales como: nombre, apellido, correo electrónico y número de teléfono.

Para insertar esos datos al sistema se utilizó el siguiente query:

```
$sentenciaSQL = "INSERT INTO contact(contact_name,
contact_last_name, contact_phone, contact_mail) VALUES ('$nombre',
'$apellido', '$telefono', '$email');";
```

3.1.12 Administración

Esta ventana permite al usuario habilitar y deshabilitar alertas ya sean de correo electrónico o de SMS tanto para un contacto en específico como para todos los contactos en general.

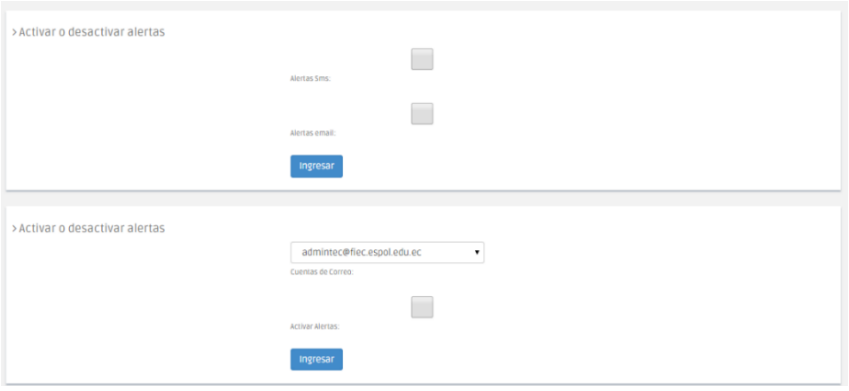


Figura 3.16: Pantalla Administración.

La figura 3.16 muestra como el usuario activa o desactiva las alertas, en la primera parte las alertas son de manera general para todos los

contactos, en cambio la segunda parte se activa o desactiva alertas a un contacto en específico.

El código se muestra en el anexo 1.

3.2 Servicio de Windows (Cliente)

El software para instalar en los computadores clientes fue desarrollado como servicio utilizando C#. El servicio trabaja en su mayor parte con las clases del sistema que son las que guardan información de componentes físicos del ordenador como procesador, discos, memorias, puertos, entre otros.

Para empezar la implementación del servicio de cliente es necesario crear un proyecto de Servicio de Windows.

En la clase Program, usando la librerías por defecto, se crea un nuevo servicio base para que este pueda iniciarse.

```
namespace Harving
{
    static class Program{
        /// Punto de entrada principal para la aplicación.
         static void Main(){
            ServiceBase[] ServicesToRun;
            ServicesToRun = new ServiceBase[]  { new Harving() };
            ServiceBase.Run(ServicesToRun);
        }
    }
}
```

3.2.1 Obtener la dirección IP del equipo

El primer dato necesitado para construir la trama que se envía al servidor es la IP del computador cliente, estos clientes pueden contener más de una dirección IP por lo que el programa primero consulta cual es la dirección IP habilitada y luego la extraerá. Esto lo logra consultando la clase *Win32_NetworkAdapterConfiguration* [12] del sistema que es la que almacena información relacionada con la configuración del adaptador de red.

```
//Buscando IP del sistema
                foreach (ManagementObject mo in moc){
                if ((bool)mo["IPEnabled"]){
```

```
                    try{
                        string[] ips = (string[])mo["IPAddress"];
// Extrayendo la IP del Sistema
                        ipSubnet[0] = ips[0].ToString();
                        break;
                    }
                    catch (Exception ex){
                        Console.WriteLine(ex.ToString());
      }
   }
}
```

3.2.2 Obtener la temperatura del equipo

Seguido de la IP el servicio buscará la temperatura del procesador. Utilizando la variable *srtemp* tipo objeto se logra conectar a la clase del sistema *MSAcpi_ThermalZoneTemperature*, con dicha variable y el lazo *foreach* se recorre todos los parámetros de la clase hasta obtener *CurrentTemperature* y hacer la conversión respectiva.

```
//Buscando Temperatura interna del CPU
foreach (ManagementObject obj1 in srtemp.Get()) {
   temp = Convert.ToDouble(obj1["CurrentTemperature"].ToString());
   temp = (temp - 2732) / 10.0;
}
tempaux = (int) temp;
```

3.2.3 Obtener el estado del mouse del equipo

Conectándose a la clase *Win32_PointingDevice* [13], por medio de la variable tipo objeto *srmouse* y recorriendo la misma con *foreach*, se obtiene el parámetro *STATUS* que nos dice si el mouse del equipo se encuentra conectado o no.

```
//Buscando el estado del mouse
foreach (ManagementObject obj3 in srmouse.Get()) {
   smouse = (String)obj3["Status"];
   if (string.Compare(smouse, OK) == 0) statem++;
}
```

3.2.4 Obtener el estado del teclado del equipo

El último campo que forma parte del paquete es el estado del teclado. Leyendo la información de la clase *Win32_Keyboard* [14], por medio del objeto *srkeyboard* creado previamente, encontramos dicha información.

```
//Buscando el estado del Teclado
foreach (ManagementObject obj2 in srkeyboard.Get()) {
   skeyboard = (String)obj2["Status"];
   if (string.Compare(skeyboard, OK) == 0) statek++;
}
```

3.2.5 Paquete de información

El servidor interpreta tramas con el formato de la figura 3.15 por lo que con la información almacenada en las variables ipSubnet[0], tempaux, statem y statek construirá un paquete con una variable de nombre data tipo String que almacena la información a ser enviada.

```
//Concatenando ipLocal con temperatura
 data = string.Concat(ipSubnet[0], ':');
 data = string.Concat(data, tempaux.ToString());

 //Concatenando ipLocal+temperatura con estadoMouse
   data = string.Concat(data, ':');
   if (statem > 0) data = string.Concat(data, OK);
   else data = string.Concat(data, NO);

//Concatenando ipLocal+temperatura+estadoMouse con estadoKeyboard
   data = string.Concat(data, ':');
   if (statek > 0) data = string.Concat(data, OK);
   else data = string.Concat(data, NO);
```

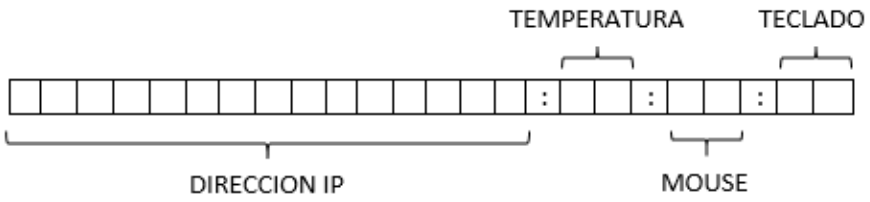


Figura 3.17: Trama que envía el cliente

3.2.6 Envío del paquete

Para el envío de información se utiliza *WebSockets* que es la tecnología que ofrece C# y que también entienda PHP instalado en el servidor. Para ello el servidor se encuentra escuchando peticiones de conexión en la red por el puerto 4455 y así los equipos que cuenten con el servicio instalado soliciten realizar una conexión *WebSockets* y establecer comunicación.

```
Socket my_Socket = new Socket(AddressFamily.InterNetwork,
SocketType.Stream, ProtocolType.Tcp);
IPEndPoint my_server = new
IPEndPoint(IPAddress.Parse("200.9.176.13"), 4455);
try {
   my_Socket.Connect(my_server);
   paquete = Encoding.ASCII.GetBytes(data);
   my_Socket.Send(paquete);
   my_Socket.Close();
               }
catch (Exception error) {
   Console.WriteLine("Error: {0}",     error.ToString());
               }
```

3.3 Servidor

El servidor fue desarrollado en PHP y fue dividido en dos scripts, *socket_server* atiende las peticiones que se encuentran enviando los ordenadores clientes y *alarm_server* se mantiene analizando las alarmas que se almacenan en la tabla *alarm*.

3.3.1 Preparación del servidor

El servidor, que trabaja en CentOS7, necesita tener instalado Postgresql y PHP para lo cual ejecutamos las siguientes líneas de comandos:

```
Instalar y configurar Postgresql:
rpm -Uvh http://yum.postgresql.org/9.4/redhat/rhel-7-x86_64/pgdg-
centos94-9.4-1.noarch.rpm
yum update
yum install postgresql94-server postgresql94-contrib
/usr/pgsql-9.4/bin/postgresql94-setup initdb
systemctl enable postgresql-9.4
systemctl start postgresql-9.4
su – postgres
$createdb harvingdb
$ createuser -P -s -e harving
$psql
postgres=# grant all privileges on database harvingdb to harving;

Instalando PHP y complementos:
yum -y install php php-mysql php-pgsql
yum -y install mod_perl mod_wsgi mod_ssl
```

3.3.2 Script socket server

El script *socket_server* no solo se encarga de recibir los paquetes enviados por los ordenadores sino que los analiza para comprobar que los parámetros de los equipos sean los esperados, en caso de no serlos genera una alarma en la tabla correspondiente. El script está divido en cuatro funciones: *receive_sockets, save_info_pc, getStates_before, save_alarm.*

La función **receive_sockets** recibe los paquetes de los equipos que cuentan con el servicio cliente, usa la función stream_socket_accept para permitir a los dispositivos que utilizan WebSockets establecer una comunicación con el servidor. Esta función trabaja en conjunto con save_info_pc para guardar los datos en la tabla machine que lleva la información de todos los computadores.

```
Function receive_sockets($ipServer,$portNumber,$nbSecondsIdle){
        // creating the socket...
        $socket =
stream_socket_server('tcp://'.$ipServer.':'.$portNumber, $errno,
$errstr);
        if (!$socket){
                echo "$errstr ($errno)<br />\n";
        }
        else {
        // while there is connection, i'll receive it... if I
didn't receive a message within $nbSecondsIdle seconds, the
following function will stop.
        while ($conn =
@stream_socket_accept($socket,$nbSecondsIdle)) {
                $message= fread($conn, 24);
                echo 'I have received that : '.$message;
                list($ip_in, $temp_in, $mouse_in,
$key_in)=explode(':',$message);
                save_info_pc($ip_in, $temp_in, $mouse_in,
$key_in);
                fputs ($conn, "OK\n");
                fclose ($conn);
        }
    fclose($socket);
  }
}
```

La función **save_info_pc** Contiene el código que actualiza la información que recibe de los equipos clientes y la analiza, parámetro por parámetro, por si debe generar una alarma. Funciona en conjunto con *getStates_before* y envía el siguiente query que almacena los datos.

Estructura de la función:

```
function save_info_pc($ip, $temp, $mouse, $keyboard){}
```

Query que emplea:

```
$query = "UPDATE machine SET mch_temp='$temp', mch_mouse='$mouse',
mch_keyboard='$keyboard' WHERE mch_ip='$ip'";
```

La función **getStates_before** ayuda a saber si se debe generar una alarma, conociendo el estado anterior de un periférico y comparándolo con el recibido en la trama puede crear un nuevo registro en la tabla *alarm*.

La función **save_alarm** guarda las alarmas que se generan mientras se encuentra atendiendo las conexiones de los clientes. Recibe los parámetros IP, tipo y descripción para almacenarlos en la tabla *alarm*.

```
function save_alarm($ip, $tipo, $descripcion){
        $dbconn = pg_connect("host=200.9.176.13 dbname=harvingdb
user=harving password=admin123")
        or die('No se ha podido conectar: ' . pg_last_error());
$query = "INSERT INTO alarm (alarm_ip, alarm_type, alarm_desc,
alarm_sent) VALUES ('$ip', '$tipo', '$descripcion', 'NO')";
        $result = pg_query($query) or die('La consulta fallo: ' .
pg_last_error());
}
```

3.3.3 Script alarm server

El script *alarm_server* está monitoreando si existen nuevas alarmas en la tabla *alarm*, en el caso que existan se genera un correo y un mensaje de texto a los contactos habilitados. Este script está formado por la función *scan_alarm* que trabaja en conjunto con las funciones *send_mail* y *send_sms.*

La función **scan_alarm** monitorea nuevas alarmas consultando el campo alarm_sent que indica si la alarma guardada ya ha sido enviada.

La función *scan_alarm* utiliza los siguientes querys para: obtener las alarmas que no han sido atendidas, los contactos a los que se deben enviar y saber si se debe emitir un SMS, un correo o ambos.

```
SELECT * FROM alarm WHERE alarm_sent='NO'

SELECT contact_phone, contact_mail FROM contact WHERE
contact_status='ON'

SELECT alarm_status_on_off FROM alarm_status WHERE
alarm_status_type='MAIL'

SELECT alarm_status_on_off FROM alarm_status WHERE
alarm_status_type='SMS'

 UPDATE alarm SET alarm_sent='SI' WHERE alarm_id=$id
```

La función **send_mail** se encarga de crear el correo electrónico al contacto almacenado y habilitado en la base de datos. Utilizando una cuenta de correos de FIEC y conectándose al servidor de correos de la facultad envía la alerta con los parámetros recibidos de la tabla de alarmas.

La función **send_sms** genera una alerta con los valores almacenados en los campos en la tabla correspondiente pero por mensaje de texto. Utilizando una cuenta de *TextMagic* que permite enviar mensajes por internet.

3.4 Sensor de temperatura y humedad en el centro de datos

Se utiliza un sensor de temperatura y humedad dht11 tal que los pines vcc y gnd los cuales son conectados a los pines de alimentación 5v y gnd, y el pin de datos fue conectado a la entrada digital 2 del arduino.

Para obtener la temperatura y humedad se utilizaron las siguientes funciones de la librería dht instalada previamente:

```
int h = dht.readHumidity();// Lee la humedad
int t= dht.readTemperature();//Lee la temperatura
```

Al ejecutar dichas funciones la temperatura y humedad de la sala de datos se almacenan en las variables t y h respectivamente, para el cual, después de

realizar las mediciones correspondientes, envía los datos al servidor para que éste ejecute un script en PHP llamado *datosSala.php* que recibe mediante método GET las mediciones y las almacena en la base de datos, a continuación se muestra como se realizó dicha operación:

```
String sentencia = "GET /datosSala.php?temp=";
sentencia += t;
sentencia += "&hum=";
sentencia += h;
Serial.println(sentencia);
client.print(sentencia); // Enviamos los datos por GET
client.println(" HTTP/1.0");}
client.println("User-Agent: Arduino 1.0");
client.println();
```

Después de enviar los datos desde el *Arduino* el servidor ejecuta un script PHP que recibe los datos y hace las comparaciones si existe una irregularidad, el script ejecuta una función save_alarm() tal que genera la alarma y la almacena para así que esta sea enviada mediante correo electrónico y SMS dependiendo de cómo se ha configurado el script, tal como se lo muestra a continuación:

```
include_once 'conexion.php';
$temperatura = $_GET['temp'];
$humedad =  $_GET['hum'];
$sentenciaSQL = "INSERT INTO
serverroom(servers_temperature,servers_humidity) VALUES
('$temperatura','$humedad')";
$resultado = pg_query($sentenciaSQL) or die('La consulta fallo: ' .
pg_last_error());
if($temperatura > 26){
save_alarm('RACK', 'TEMPERATURE', 'TEMPERATURA ALTA');
}
if($humedad > 49){
save_alarm('RACK', 'HUMIDITY', 'HUMEDAD ALTA'
);
}
function save_alarm($ip, $tipo, $descripcion){
   // Conectando y seleccionado la base de datos
         $dbconn = pg_connect("host=$ip_server dbname=$dbname
user=harving password=$userpass")
         or die('No se ha podido conectar: ' . pg_last_error());

   $query = "INSERT INTO alarm (alarm_ip, alarm_type, alarm_desc,
alarm_sent) VALUES ('$ip', '$tipo', '$descripcion', 'NO')";
         $result = pg_query($query) or die('La consulta fallo: ' .
pg_last_error());
}
```

CAPÍTULO 4

4. PRUEBAS Y RESULTADOS.

Se desarrolló un sistema antirrobo y control de temperatura de equipos de cómputo que facilita el monitoreo de periféricos de computadoras y estado de la sala de rack que agiliza el trabajo no solo de los asistentes sino también de los ayudantes.

El servicio cliente es capaz de obtener la información que se necesita monitorear y enviarla al servidor para ser analizada. El servidor recibe los datos enviados por los clientes, los almacena y genera las alertas correspondientes. Se puede configurar la manera de alertar a un asistente o ayudante, ya sea mediante mensajes de texto o correos para garantizar que la alerta sea atendida lo más pronto posible.

4.1 Pruebas de comunicación

Para comprobar la comunicación entre el servidor y los dispositivos clientes se muestran los paquetes recibidos de los computadores.

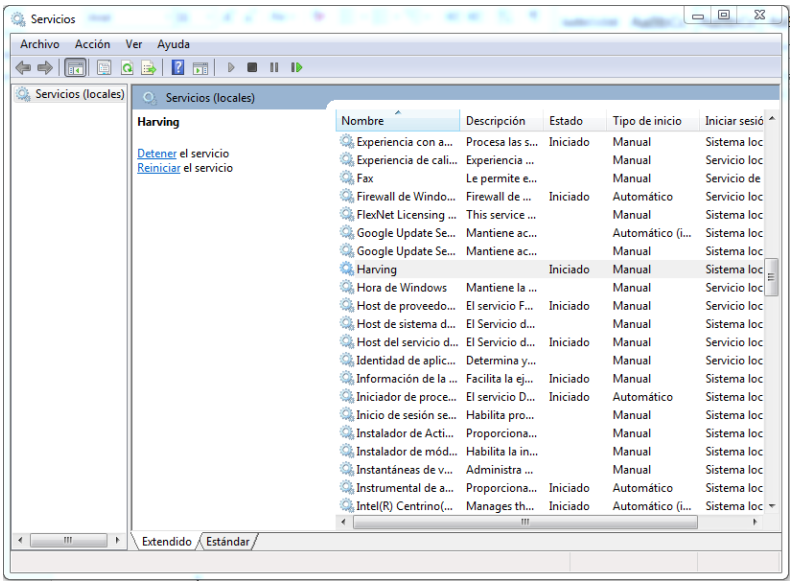


Figura 4.1: Inicialización del servicio en el cliente

Una vez iniciado el servicio y con el servidor atendiendo las solicitudes comienzan a llegar los paquetes transmitidos.

```
[admin@localhost server]$ php socket_server.php
I have received that : 200.9.176.167:29:OK:OK
```

Figura 4.2: Paquete de datos recibido por el servidor.

En la figura 4.2 se observa un paquete recibido por el servidor donde los datos están separados por dos puntos con lo cual se puede saber que la dirección IP es 200.9.176.167, la temperatura es de 29 grados centígrados, el estado del mouse es OK y el del teclado también es OK.

4.2 Pruebas de periféricos, temperatura y humedad

4.2.1 Prueba de mouse

Para la prueba de monitoreo de mouse se desconectó y volvió a conectar el periférico para observar el cambio de estado.

Figura 4.3: Desconexión del mouse.

```
[admin@localhost server]$ php socket_server.php
I have received that : 200.9.176.167:29:NO:OK
```

Figura 4.4: Paquete datos recibido por el servidor.

En la figura 4.4 llega un nuevo paquete, en el cual se observa que el estado del mouse es NO, lo que significa que el periférico ha sido desconectado.

Figura 4.5: Alerta de correo generada por el mouse.

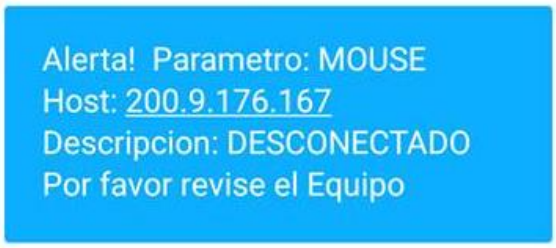


Figura 4.6: Alerta de mensaje de texto generada por el mouse.

En la figura 4.5 observamos la alerta de correo que se originó por la desconexión del mouse y en la figura 4.6 se encuentra la alerta de mensaje de texto generada por el mismo dispositivo.

Cuando volvemos a conectar el periférico se genera otra alerta.

Figura 4.7: Conectando el mouse.

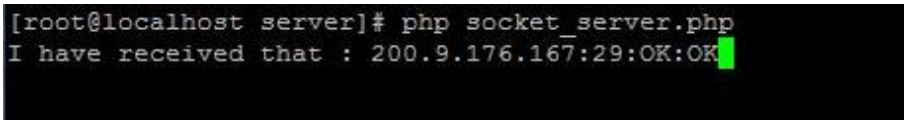


Figura 4.8: Nuevo paquete de datos recibido por el servidor.

En la figura 4.8 se muestra que el estado del mouse regresa a la normalidad.

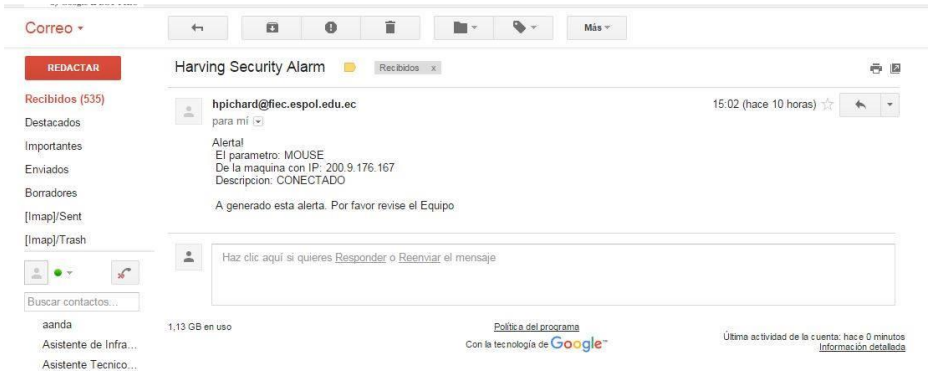


Figura 4.9: Alerta de correo generada por el mouse.

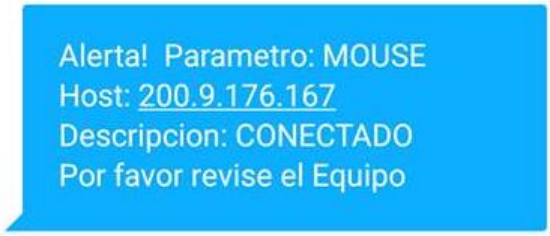


Figura 4.10: Alerta de mensaje de texto generada por el mouse.

Luego que el estado vuelve a la normalidad se genera otra alerta que muestra el nuevo estado.

4.2.2 Prueba de teclado

Igual que para la prueba del mouse, el teclado se desconectó y volvió a conectar para observar los cambios de estado.

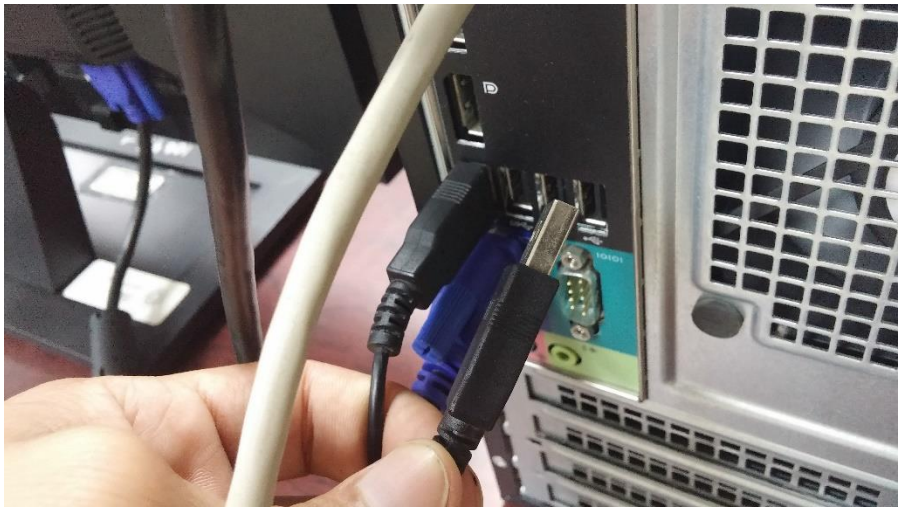


Figura 4.11: Desconectando el teclado.

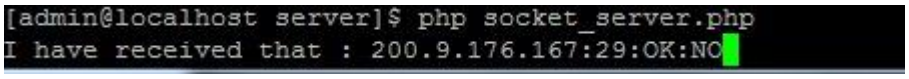


Figura 4.12: Nuevo paquete de datos recibido por el servidor.

En la figura 4.12 llega un nuevo paquete, en el cual se observa que el estado del teclado es NO, lo que significa que el periférico ha sido desconectado.

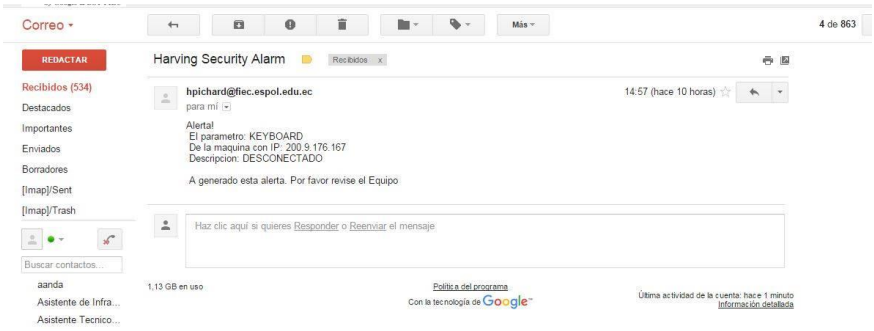


Figura 4.13: Alerta de correo generada por el teclado.

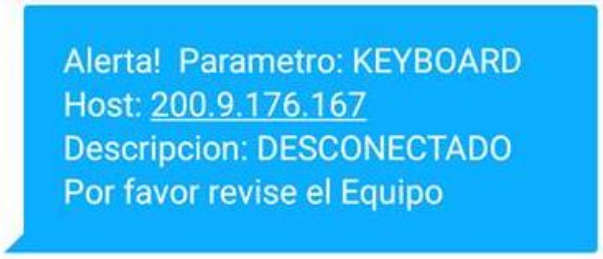


Figura 4.14: Alerta de mensaje de texto generada por el teclado.

En la figura 4.13 observamos la alerta de correo que se originó por la desconexión del teclado y en la figura 4.14 se encuentra la alerta de mensaje de texto generada por el mismo dispositivo.

Cuando volvemos a conectar el periférico se genera otra alerta.

Figura 4.15: Conectando el teclado.

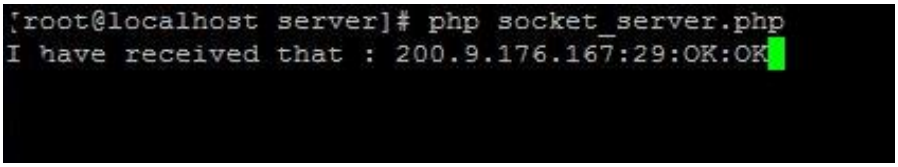


Figura 4.16: Nuevo paquete de datos recibido por el servidor.

En la figura 4.16 se muestra que el estado del teclado regresa a la normalidad.

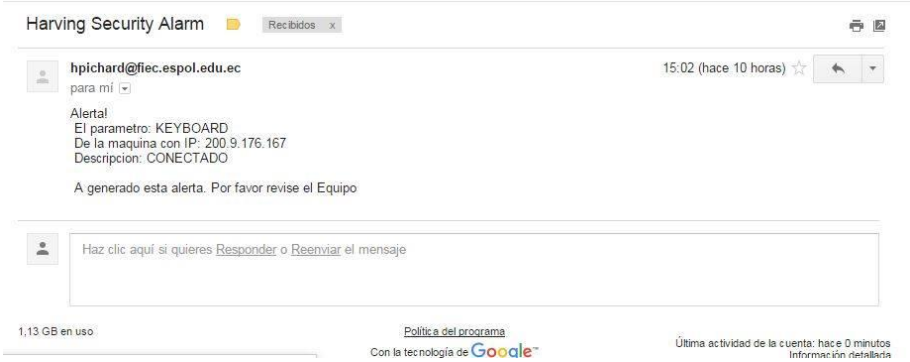


Figura 4.17: Alerta de correo generada por el teclado.

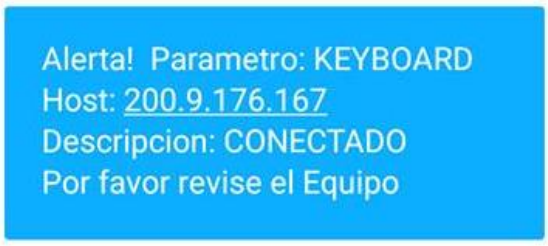


Figura 4.18: Alerta de mensaje de texto generada por el teclado.

Luego que el estado vuelve a la normalidad se genera otra alerta que muestra el nuevo estado.

4.2.3 Prueba de temperatura del ordenador

Para comprobar que se generaran alertas por temperatura se disminuyó el umbral, en este caso se configuro un umbral de 29 grados centígrados y se pueda generar una alarma.

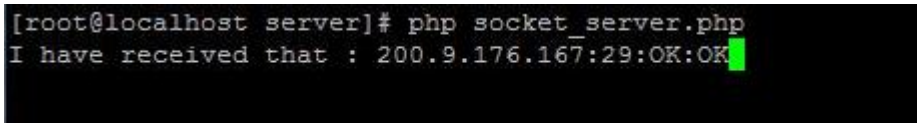


Figura 4.19: Nuevo paquete de datos recibido por el servidor.

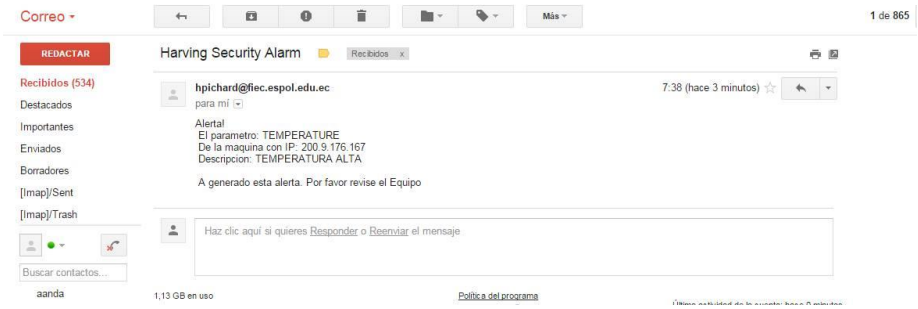


Figura 4.20: Alerta de correo generada por temperatura elevada.

Figura 4.21: Alerta de mensaje de texto generada por temperatura elevada.

En la figura 4.20 observamos la alerta de correo que se originó por la temperatura elevada en el computador cliente y en la figura 4.21 se encuentra la alerta de mensaje de texto generada por el mismo motivo.

4.2.4 Prueba de temperatura y humedad del centro de datos

Igual que con la temperatura del ordenador cliente, se bajó el umbral de las alertas de temperatura y humedad a 27 grados centígrados y a un 57 por ciento de humedad para poder generar alarmas.

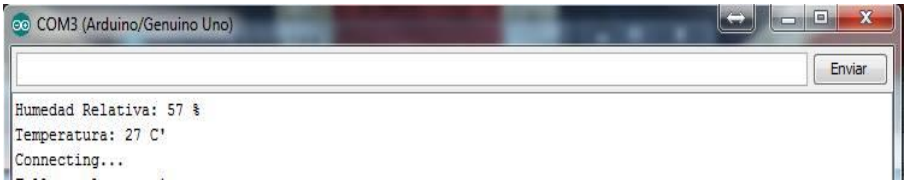


Figura 4.22: Humedad y temperatura censada por el *Arduino*

Figura 4.23: Alerta de correo generada por temperatura o humedad elevada en el centro de datos.

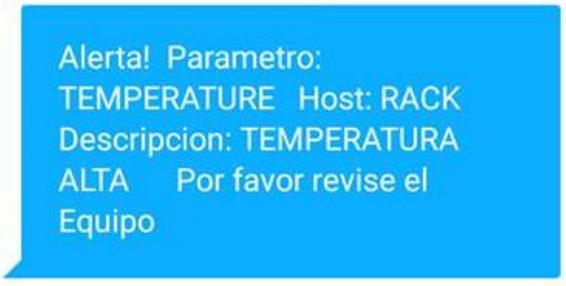


Figura 4.24: Alerta de mensaje de texto generada por temperatura o humedad elevada en el centro de datos.

En la figura 4.23 es una alarma de correo generada por la temperatura o humedad elevada en el centro de datos mientras que en la figura 4.24 se observa una alerta de mensaje de texto por el mismo motivo.

CONCLUSIONES Y RECOMENDACIONES

Conclusiones

1. Se logró implementar un sistema cliente-servidor para el monitoreo de equipos usando lenguajes de programación tales como PHP, C#, entre otros.
2. Haciendo uso de múltiples lenguajes de programación se desarrolló una interfaz en la cual los asistentes y jefes de los laboratorios de computación de la FIEC puedan ver el nivel de temperatura, estado del mouse y estado del teclado de los ordenadores que se están monitoreando.
3. Con el uso de una plataforma de desarrollo de hardware se implementó un módulo electrónico que cense la temperatura y humedad del centro de datos para luego almacenar dicha información en la base de datos.
4. Se desarrolló un script que permite enviar alertas SMS y de correo electrónico sobre elevación de temperatura o extracción de periféricos en las computadoras de los laboratorios usando API's y funciones PHP específicas de cada tipo de alerta.
5. Fue posible implementar una solución para el problema planteado anteriormente de manera eficaz y a un costo manejable para cualquier empresa que desee realizar este tipo de monitoreo.

Recomendaciones

1. Al momento de configurar el tiempo de espera para envió de paquete del cliente se recomienda utilizar como mínimo 2 minutos ya que si se le configura un tiempo muy corto, este puede llegar a saturar la red o saturar el servidor que está escuchando estos paquetes.
2. Configurar que el sistema envié solo 1 paquete de ping a las maquinas caso contrario, el sitio web se vuelve lento al momento de mostrar los ordenadores que pertenecen a un laboratorio en específico.

3. Cifrar las contraseñas en la base de datos ya que si algún intruso llegue a evadir la seguridad del servidor y acceda la base, puede obtener las credenciales del administrador del sistema.
4. Ubicar el sensor en la zona alta en el centro de datos para así poder lograr una captura de datos más precisa y con menos margen de error.
5. Debido a que la placa no realiza instrucciones complejas se recomienda utilizar un arduino de menor gama tal como el NANO para poder gastar menos dinero en la implementación.

BIBLIOGRAFÍA

[1] R. Data, «Introduction to HTML,» Refnes Data, 1999. [En línea]. Available: http://www.w3schools.com/html/html_intro.asp. [Último acceso: 9 ENERO 2016].

[2] J. M. Lazaro, «Desarrollo Web,» 01 Enero 2001. [En línea]. Available: http://www.desarrolloweb.com/articulos/26.php. [Último acceso: 16 Enero 2016].

[3] F. David, de *JavaScript: The Definitive Guide*, O'Reilly Media, 2002.

[4] D. Cochran, de *Twitter Bootstrap Web Development*, Packt Publishing, 2012.

[5] E. F. Cases, «ibrugor,» 21 Octubre 2014. [En línea]. Available: http://www.ibrugor.com/blog/que-es-php-para-que-sirve/. [Último acceso: 17 Enero 2016].

[6] J. Kovacs, «James Kovacs Weblog,» 7 Septiembre 2007. [En línea]. Available: http://jameskovacs.com/2007/09/07/cnet-history-lesson/. [Último acceso: 17 Enero 2016].

[7] rafaelma, «Postgresql-es,» 02 Octubre 2010. [En línea]. Available: http://www.postgresql.org.es/sobre_postgresql. [Último acceso: 17 Enero 2016].

[8] A. G. González, «Panama Hitek,» 20 Mayo 2015. [En línea]. Available: http://panamahitek.com/que-es-arduino-y-para-que-se-utiliza/. [Último acceso: 17 Enero 2016].

[9] I. Electrónica, «I+D Electrónica,» 2011. [En línea]. Available: http://www.didacticaselectronicas.com/index.php?page=shop.product_details&category_id=43&flypage=flypage.tpl&product_id=1279&option=com_virtuemart&Itemid=133&vmcchk=1&Itemid=133. [Último acceso: 17 Enero 2016].

[10] Arduino, «Arduino,» 2007. [En línea]. Available: https://www.arduino.cc/en/Main/ArduinoEthernetShield. [Último acceso: 17 Enero 2016].

[11] textmagic, «Textmagic api,» 2015. [En línea]. Available: https://www.textmagic.com/. [Último acceso: 15 Enero 2016].

[12] Microsoft, «Microsoft,» [En línea]. Available: https://msdn.microsoft.com/en-us/library/windows/desktop/aa394217(v=vs.85).aspx. [Último acceso: 19 Enero 2016`].

[13] Microsoft, «Microsoft,» [En línea]. Available: https://msdn.microsoft.com/en-us/library/windows/desktop/aa394356(v=vs.85).aspx. [Último acceso: 19 Enero 2016].

[14] Microsoft, «Microsoft,» [En línea]. Available: https://msdn.microsoft.com/en-us/library/windows/desktop/aa394166(v=vs.85).aspx. [Último acceso: 19 Enero 2016].

ANEXOS

Anexo 1:

$numero[] es una matriz que corresponde al valor que toma los checkboxs y $cont corresponde a cuantos checkboxs han sido activados.

```
if ($conta == 1 and $numero[0] == 1){
 echo "desactive email";
 $sentenciaSQL = "UPDATE alarm_status SET
alarm_status_on_off='OFF' WHERE alarm_status_id='1';";
 $sentenciaSQL2 = "UPDATE alarm_status SET
alarm_status_on_off='ON' WHERE alarm_status_id='2';";
 $resultado = pg_query($sentenciaSQL) or die('La consulta fallo: '
. pg_last_error());
 $resultado2 = pg_query($sentenciaSQL2) or die('La consulta fallo:
' . pg_last_error());
          }
if ($conta == 1 and $numero[0] == 2){
 echo "desactive sms";
 $sentenciaSQL = "UPDATE alarm_status SET
alarm_status_on_off='OFF' WHERE alarm_status_id='2';";
 $sentenciaSQL2 = "UPDATE alarm_status SET
alarm_status_on_off='ON' WHERE alarm_status_id='1';";
 $resultado = pg_query($sentenciaSQL) or die('La consulta fallo: '
. pg_last_error());
 $resultado2 = pg_query($sentenciaSQL2) or die('La consulta fallo:
' . pg_last_error());
          }
 if ($conta == 2){
 $sentenciaSQL1 = "UPDATE alarm_status SET
alarm_status_on_off='ON' WHERE alarm_status_id='2';";
 $sentenciaSQL2 = "UPDATE alarm_status SET
alarm_status_on_off='ON' WHERE alarm_status_id='1';";
 $resultado = pg_query($sentenciaSQL1) or die('La consulta fallo:
' . pg_last_error());
 $resultado2 = pg_query($sentenciaSQL2) or die('La consulta fallo:
' . pg_last_error());

          }
```

En el primer if se tomó en cuenta cuando está activo solo el de las alertas sms por el que se ejecuta el query respectivo.

El segundo if se tomó en cuenta cuando está activo solo el de las alertas mail por el que se ejecuta el query respectivo.

El tercer if se tomó en cuenta cuando están activo tanto las alertas sms como las mail por el que se ejecuta el query respectivo.

El código se muestra en el anexo 1 donde $numero[] es una matriz que corresponde al valor que toma los checkboxs y $cont corresponde a cuantos checkboxs han sido activados.