ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

Facultad de Ingeniería en Electricidad y Computación

"CARRETERA INTELIGENTE CON MENSAJES DE ADVERTENCIAS Y CIERRE DE VÍAS DEBIDO A LA AFECTACIÓN DE CONDICIONES INESPERADAS"

INFORME DE MATERIA INTEGRADORA

Previa a la obtención del Título de:

INGENIERO/A EN TELEMÁTICA

REINA TORRES KIARA CAROLINA

MONTENEGRO VIERA CARLOS LUIS

GUAYAQUIL – ECUADOR

AÑO: 2016

AGRADECIMIENTOS

Mis más sinceros agradecimientos a Dios porque ha estado conmigo a cada paso que doy, cuidándome y dándome fortaleza para continuar. A mi compañero de tesis porque en esta armonía grupal lo hemos logrado y a mi director de tesis quién nos ayudó en todo momento, Ing. Washington Velásquez.

KIARA CAROLINA REINA TORRES

Agradezco primeramente a Dios porque me ha guiado a lo largo de estos años de estudio; a mis padres y mis tres hermanos por ese apoyo incondicional y enseñanzas que me permitieron alcanzar este objetivo; a mi familia en general; y al Ing. Washington Velásquez por el apoyo brindado en el desarrollo de este proyecto.

CARLOS LUIS MONTENEGRO VIERA

DEDICATORIA

La concepción de este proyecto está dedicada a mis padres, pilares fundamentales en mi vida. Sin ellos, jamás hubiese podido conseguir lo que hasta ahora. Su tenacidad y lucha insaciable han hecho de ellos el gran ejemplo a seguir y destacar, a mi esposo, compañero inseparable de cada jornada. El representó gran esfuerzo y tesón en momentos de decline y cansancio.

Y finalmente pero no menos importantes, a mis hijos, Arleth y Aaron, ellos han representado mi motor de lucha diaria, e inspiración, demostrándoles que nunca hay que rendirse, y que no hay obstáculo que no pueda ser superado.

A todos ellos este proyecto, que sin ellos, no hubiese podido ser, los amo con mi vida.

KIARA CAROLINA REINA TORRES

DEDICATORIA

Dedicado a mis padres Efrén y Adriana porque son la motivación que me impulsa a cumplir las metas propuestas; porque sembraron buenos valores en mí y me brindaron la educación necesaria para formarme como profesional y sobre todo como persona.

A mis hermanos Efrén, Adriana y Alvaro; que han estado conmigo en todo momento dándome su apoyo incondicional.

A mi abuela Ángela, quien ha estado siempre junto a mí, aconsejándome y brindándome su apoyo y afecto.

CARLOS LUIS MONTENEGRO VIERA

TRIBUNAL DE EVALUACIÓN

Ing. Washington Velásquez, MSc.

PROFESOR EVALUADOR

Ing. Adriana Collaguazo, 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"

.....

Kiara Carolina Reina Torres

.....

Carlos Luis Montenegro Viera

RESUMEN

En base a la problemática que existe en las carreteras del Ecuador; las cuales no cuenta con un sistema que controle e informe sobre acontecimientos ocurridos tales como: derrumbes, accidentes de tránsito, marchas, entre otros. Se implementó un módulo que, con ayuda de diversas tecnologías, se logra que una carretera se convierta en inteligente.

Se implementó un Servidor Web y de Base de Datos; donde se aloja la página web que le permite al administrador del sistema realizar el cierre o apertura de vías según sea el caso; además de una base de datos relacional donde está almacenada información de los usuarios, las carreteras y de los sucesos que en ellas ocurren. También se situó un módulo en la carretera que monitorea permanentemente la carretera en busca de obstáculos, y realiza el envío respectivo de alertas en caso de hallar alguno.

Se utilizó la plataforma electrónica Arduino Uno R3 la cual se encarga de tomar los datos leídos por lo sensores y procesarlos así permite clasificar los acontecimientos en la carretera para posteriormente enviar los mensajes de advertencia (envío automático de un correo electrónico y mensaje de texto).

Para la detección de obstáculos utilizamos sensores ultrasónicos los cuales toman lectura de las distancias del obstáculo, las almacenan, se validan y finalmente se clasifica el suceso de acuerdo a qué tanto esté obstruyendo la vía; estas lecturas se envían al servidor para que las almacene en la base de datos, y posteriormente hacer él envió de las alertas: por correo electrónico se configuró la cuenta Gmail y para el envío del mensaje SMS usamos el API Sinch SMS.

Finalmente se ha desarrollado un sistema de video vigilancia utilizando la Raspberry Pi 2. De esta manera pudimos acceder al video en vivo simplemente realizando una petición ingresando a la dirección en el navegador predeterminada.

INDICE GENERAL

INTRODUCCIÓN

Debido a las diferentes causas de obstrucción de carreteras o cierre de vías, provocan que viajeros y conductores tengan que desviarse hacia nuevas rutas o tomar caminos más largos para llegar a su destino poniendo en riesgo la seguridad, provocando un consumo de tiempo innecesario al momento de viajar.

Por ello es necesario realizar un monitoreo constante de las carreteras para conocer el momento exacto en que ocurrió algún acontecimiento y tomar alguna acción que permita informar al usuario del estado de la vía. Con ello el usuario puede realizar una mejor planificación de su recorrido y tomar rutas alternativas que le permitan llegar a su destino.

Con la ayuda que nos proporciona la tecnología actual se puede comprobar algún bloqueo en la carretera, tales como derrumbes, accidentes, marchas, entre otros; y esta información procesarla y enviarla en los diferentes tipos de alerta que disponemos para darle aviso al administrador, tales como correo electrónico, SMS y la opción de ingresar directamente a la página web y monitorear los acontecimientos de las carreteras.

CAPÍTULO 1

1. GENERALIDADES

1.1 Antecedentes

En las carreteras del Ecuador existe señalización de caminos, pero no se cuenta con algún sistema que controle e informe sobre fenómenos naturales ocurridos tales como: derrumbes, accidentes de tránsito, marchas, entre otros, debido a que estos eventos ocurren sin previo aviso produciendo obstrucción de vías, y hasta accidentes de tránsito.

En una vía montañosa pueden existir derrumbes, lo cual afecta la seguridad y la vida de quienes transitan la vía; así como también ocasionan malestares dependiendo de la magnitud del mismo, lo cual ocasiona una pérdida de tiempo considerable y provocan que estos deban retornar o tomar nuevas vías para llegar a su destino.

Muchos de los fenómenos naturales, ocasionan accidentes que pueden producir tráfico de vías, o en el más crítico de los casos pérdidas irremediables como la vida de una persona si no recibe la ayuda necesaria a tiempo.

Debe existir un sistema informativo eficiente, que permita conocer los sucesos adversos de una carretera y logre llegar a usuarios; evitando accidentes, tráfico vial y mejore el desempeño a cada grupo de trabajo que interviene en estos acontecimientos.

1.2 Descripción y análisis del problema

Hoy en día el tiempo es un factor muy importante, y un recurso necesario el cual desaprovecharlo no debería ser una opción, aunque muchas veces sea por problemas que se encuentran fuera del alcance de nuestras manos; como por ejemplo el movilizarnos de un punto a otro puede conllevar muchos contratiempos al estar expuestos a un sin número de inconvenientes (tales como derrumbes, marchas, fuertes lluvias, accidentes de tránsito, entre otros) que provocan tráfico en las carreteras o vías, retrasos y pérdida de tiempo.

Por ello, se presenta la implementación de un módulo de carreteras inteligente, el cual permite el envío de notificaciones: por correo y mediante un mensaje de texto; al administrador del sistema, que se encarga de alertar a los ciudadanos sobre el cierre de las vías afectadas, después de haber detectado alguno de los inconvenientes mencionados anteriormente. Cabe recalcar que los problemas son detectados mediante el uso de sensores que se encuentran colocados a lo largo de las vías y proporcionan la señal de alerta que indica el acontecimiento.

1.3 Justificación

La necesidad de contar con un módulo que permita el monitoreo de las carreteras del país y alertar a los ciudadanos sobre posibles obstrucciones en las mismas sumado a los avances tecnológicos en lo que respecta a redes sensoriales, las cuales nos permiten obtener datos del entorno justifican el desarrollo de este prototipo. La facilidad de transportar las lecturas de los sensores a la página web, manipulada por el administrador, será importante ya que permite monitorear las carreteras de manera remota simplemente ingresando a una página web y observar en detalles las respectivas mediciones.

Es una herramienta que provee un gran beneficio para el usuario ya que le permite optimizar tiempo al poder tomar rutas alternativas para llegar a su destino, una vez que el sistema alerte sobre algún percance en la vía. Otro beneficio importante es la posibilidad de prevenir accidentes debido a fenómenos naturales (como es el caso de los derrumbes) que pueden costar vidas humanas.

1.4 Objetivos

1.4.1 Objetivo General

Implementar un sistema que convierte a una carretera en inteligente, el cual consiste en monitorear y alertar a la guardia zonal sobre la existencia de derrumbes u otro tipo de riesgos mediante el uso de diversas tecnologías; generando un envío automático de mensajes de advertencia, utilizando plataformas de hardware libre.

1.4.2 Objetivos Específicos

- Implementar una red de sensores a lo largo de las carreteras monitoreadas que permitan captar información sobre obstrucciones en la vía.
- Utilizar un sensor ultrasónico de bajo costo que no sólo pueda detectar la presencia de un objeto, sino también a qué distancia se encuentra; todos ellos conectados a una placa Arduino.
- Diseñar una base de datos para el almacenamiento de la información de los acontecimientos ocurridos, utilizando un sistema de gestión bases de datos relacionales de código abierto como MySQL, basado en un lenguaje estructurado.
- Establecer una comunicación con el dispositivo móvil del administrador para alertar sobre algún problema haciendo uso de la tecnología GSM.
- Realizar el envío automático de un correo electrónico mediante el uso de una aplicación de consola en el servidor, como otra alternativa para alertar al administrador.
- Implementar un sistema de video vigilancia haciendo uso de una Raspberry Pi 2.
- Establecer comunicación de manera inalámbrica entre el módulo ubicado en la carretera y el servidor de Bases de Datos utilizando el módulo WIFI ESP8266, para almacenar información de los sucesos.

1.5 Alcance

El prototipo que se ha desarrollado realiza el monitoreo de las carreteras en un punto determinado de las mismas, buscando obstáculos que impidan el paso de los vehículos y alertando al administrador sobre lo que acontece de manera que sea él quien decida si la carretera debe ser cerrada o puede permanecer abierta.

1.6 Limitaciones

- Para el funcionamiento de este prototipo es necesario la existencia de una red inalámbrica en el lugar donde vaya a ser instalado, dado que el envío de los datos se los realiza a través de un medio no guiado.
- El rango de distancias que abarca el sensor ultrasónico utilizado nos limita un poco en el ancho que deberían tener las carreteras en las cuales se vaya a implementar el módulo.

CAPÍTULO 2

2. MARCO TEÓRICO

2.1 Sistema GSM

El Sistema GSM (Global System for Mobile communications) GSM es un sistema de telecomunicaciones digitales celulares normalizado por el Instituto Europeo para la Normalización en Telecomunicaciones (ETSI). La red de comunicaciones móviles GSM proporciona enlaces de comunicación entre usuarios del servicio de comunicaciones móviles, incluso si se encuentran en células distintas o en el dominio de diferentes operadores, así como conexiones entre usuarios del servicio de comunicaciones móviles y usuarios de las redes fijas. [1]

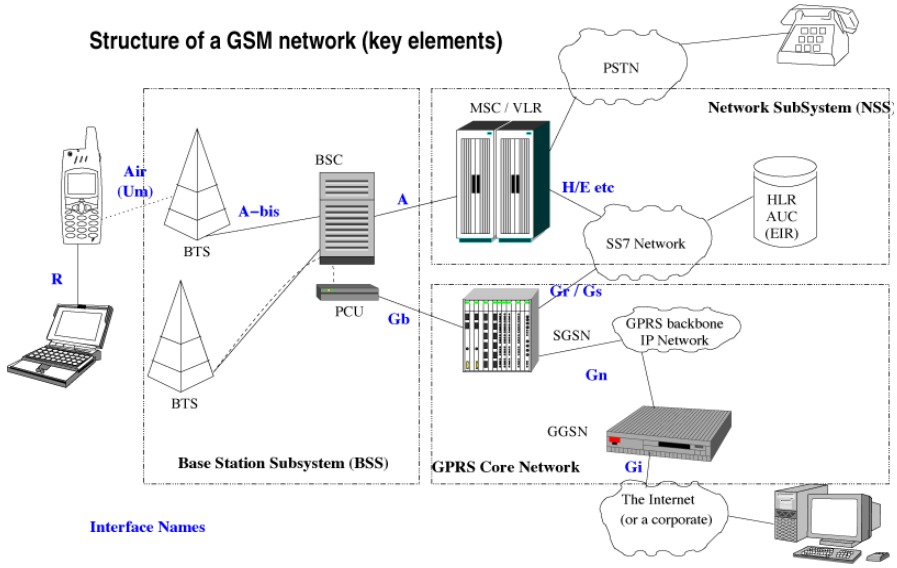

Figura 2.1. Esquema general de un sistema GSM [11]

2.2 Arduino UNO

Es una plataforma de electrónica de open-hardware y open-source, flexible y fácil de usar para la creación de prototipos**.** El lenguaje de programación de Arduino es una implementación de Wiring, que a su vez se basa en Processing, un entorno de programación multimedia. Las placas Arduino pueden medir valores ambientales al recibir información de variedad de sensores y afectar sus alrededores controlando luces, motores y otros actuadores. [4]

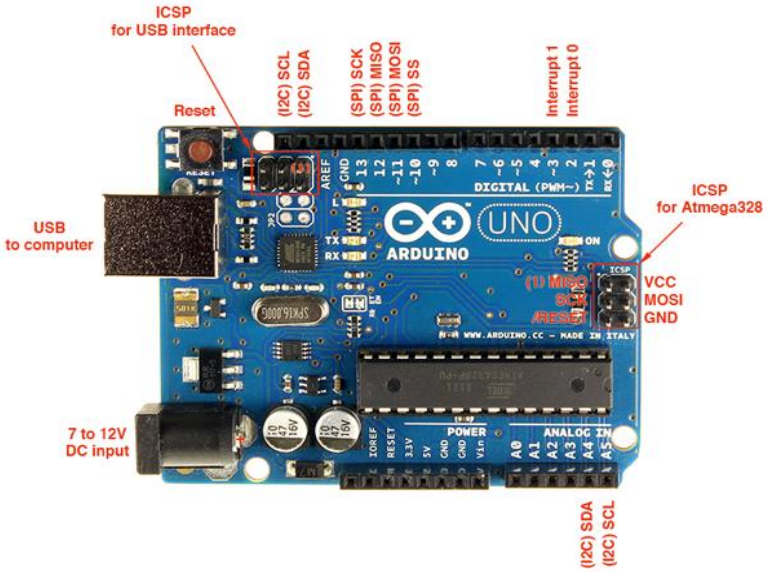

Figura 2.2. Placa Arduino [3]

2.2.1 Características de Arduino

Terminales Digitales.

Las terminales digitales de una placa Arduino pueden ser utilizados para entradas o salidas de propósito general a través de los comandos pinMode(), digitalRead(), y digitalWrite(). Cada terminal tiene una resistencia pull-up que puede activarse o desactivarse utilizando DigitalWrite() (con un valor de HIGH o LOW, respectivamente) cuando el pin está configurado como entrada. La corriente máxima por salida es 40 mA. [3]

Pines Analógicos.

Los pines de entrada analógicos soportan conversiones analógico-digital (ADC) de 10 bit utilizando la función analogRead(). Las entradas analógicas pueden ser también usadas como pines digitales: entrada analógica 0 como pin digital 14 hasta la entrada analógica 5 como pin digital 19. Las entradas analógicas 6 y 7 (presentes en el Mini y el BT) no pueden ser utilizadas como pines digitales. [3]

Pines de alimentación.

VIN (a veces marcada como "9 V"). Es el voltaje de entrada a la placa Arduino cuando se está utilizando una fuente de alimentación externa (En comparación con los 5 voltios de la conexión USB o de otra fuente de alimentación regulada). Se puede proporcionar voltaje a través de este pin. Diferentes placas aceptan distintos rangos de voltaje de entrada, todo está detallado en la documentación de la placa. El LilyPad no tiene pin VIN y acepta solo una entrada regulada. [3]

2.3 Wifi ESP8266

El módulo WiFi ESP8266 es un auto SOC contenida con la pila de protocolos TCP / IP integrado que puede dar cualquier acceso microcontrolador a tu red WiFi. Capaz de acoger bien una aplicación o la descarga de todas las funciones de redes Wi-Fi de otro procesador de aplicaciones. Cada módulo ESP8266 viene pre-programado con un comando AT conjunto firmware. [4]

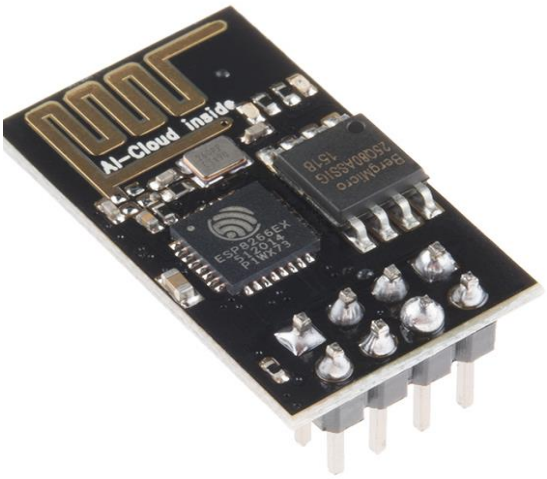

Figura 2.3. Módulo WIFI-ESP8266 [4]

2.3.1 Características

Este módulo tiene una capacidad de gran alcance de procesamiento y almacenamiento que le permite integrarse con los sensores y dispositivos específicos de la aplicación a través de sus GPIOs con un desarrollo mínimo por adelantado y la carga mínima durante el tiempo de ejecución. Su alto grado de integración en el chip permite la circuitería externa mínima, incluyendo el módulo de front-end, está diseñado para ocupar el área mínima de PCB. El ESP8266 apoya APSD para aplicaciones VoIP y las interfaces de co-Existance Bluetooth, contiene un RF auto-calibrado que permite que funcione en todas las condiciones de funcionamiento, y no requiere de partes de RF externas. [4]

2.4 Sensor HC-SR04

El HC-SR04 es un sensor ultrasónico de bajo costo que no sólo puede detectar si un objeto se presenta, como un sensor PIR (Passive Infrared Sensor), sino que también puede sentir y transmitir la distancia al objeto. [5]

Figura 2.4. Sensor HC-SR04 [12]

2.4.1 Características

- Tienen dos transductores, básicamente, un altavoz y un micrófono.
- Ofrece una excelente detección sin contacto (remoto) con elevada precisión y lecturas estables en un formato fácil de usar.
- El funcionamiento no se ve afectado por la luz solar o el material negro como telémetros ópticos (aunque acústicamente materiales suaves como telas pueden ser difíciles de detectar).
- La velocidad del sonido en el aire (a una temperatura de 20 °C) es de 343 m/s. (por cada grado centígrado que sube la temperatura, la velocidad del sonido aumenta en 0,6 m/s). [5]

2.5 WampServer

WampServer es un entorno de desarrollo web. Te permite crear aplicaciones web con Apache 2, PHP y una base de datos MySQL.Al lado, PhpMyAdmin permite administrar fácilmente sus bases de datos. [6]

2.6 MYSQL

Es un sistema de gestión de bases de datos relacional, multihilo y multiusuario. [13]

2.6.1 Principales características

- Escrito en C y en C++.
- Probado con un amplio rango de compiladores diferentes.
- Funciona en diferentes plataformas.
- Proporciona sistemas de almacenamientos transaccionales y no transaccionales.
- Un sistema de reserva de memoria muy rápido basado en threads.
- Un sistema de privilegios y contraseñas que es muy flexible y seguro, y que permite verificación basada en el host. [9]

2.7 APACHE

Apache es un servidor web flexible, rápido y eficiente, continuamente actualizado y adaptado a los nuevos protocolos (HTTP 1.1). [10]

2.7.1 Características

- Multiplataforma.
- Es un servidor de web conforme al protocolo HTTP/1.1.
- Modular: Puede ser adaptado a diferentes entornos y necesidades, con los diferentes módulos de apoyo que proporciona, y con la API de programación de módulos, para el desarrollo de módulos específicos.
- Basado en hebras en la versión 2.0.
- Incentiva la realimentación de los usuarios, obteniendo nuevas ideas, informes de fallos y parches para la solución de los mismos.
- Se desarrolla de forma abierta.
- Extensible: gracias a ser modular se han desarrollado diversas extensiones entre las que destaca PHP, un lenguaje de programación del lado del servidor.
- Puede realizar autenticación de datos utilizando SGDB.
- Soporte de seguridad SSL y TLS.

- Puede dar soporte a diferentes lenguajes, como Perl, PHP, Phyton y tcl. [9]

2.8 PHP

Es un lenguaje de código abierto muy popular adecuado para el desarrollo web, puede ser usado en la mayoría de los servidores web al igual que en casi todos los sistemas operativos y plataformas sin ningún costo.

.

2.8.1 Características

- Lenguaje interpretado.
- Puede crear contenido dinámico web y aplicaciones para servidores.
- Es posible crear aplicaciones gráficas utilizando la biblioteca gtk+.
- El código está encerrado entre las etiquetas especiales de comienzo y final <?php y?>.
- El código es ejecutado en el servidor.
- Ofrece rapidez y seguridad. [10]

CAPÍTULO 3

3. IMPLEMENTACION Y DESARROLLO

3.1 Modalidad de la investigación

Se ha ejecutado un proyecto de investigación con modalidad aplicada, debido a que se utilizó los conocimientos científicos y técnicos para brindar una solución al problema existente, utilizando la tecnología actual.

Se ha empleado una investigación bibliográfica debido a la necesidad del sustento científico para obtener la información, la cual se encontró en tesis de grado, proyectos, tutoriales y en el Internet, incrementando el conocimiento necesario para el desarrollo del presente módulo.

Además, se ha utilizado la investigación experimental puesto que se ha realizado la implementación de la solución encontrada y verificado el respectivo funcionamiento del sistema.

3.2 Plan de recolección de información

La recolección se ha realizado basada en información obtenida de sucesos ocurridos en diferentes carreteras así, se ha llegado a comprender de una manera más generalizada los fenómenos que producen un cierre de vías.

3.3 Procesamiento y análisis de datos

Una vez obtenida la información necesaria, se sometió a un análisis en base a tablas de resultados encontrando los más importantes y relevantes; considerando mayormente el factor peligro.

3.4 Desarrollo del módulo

En la implementación se deben diferenciar dos partes fundamentales, que mediante el intercambio de información permite el correcto funcionamiento de este. Una de estas partes corresponde al Servidor Web y de Base de Datos; donde se encuentra alojada la página web que le permite al administrador del sistema realizar el cierre o apertura de vías según sea el caso; además de una base de datos relacional donde almacenamos información de los usuarios del sistema, las carreteras y de los sucesos que en ellas ocurran. La segunda parte de la implementación hace referencia al módulo situado en la carretera que permite monitorear permanentemente en busca de obstáculos y realizar las respectivas alertas en caso de hallarlos. De esta manera nuestro esquema queda definido de la siguiente manera:

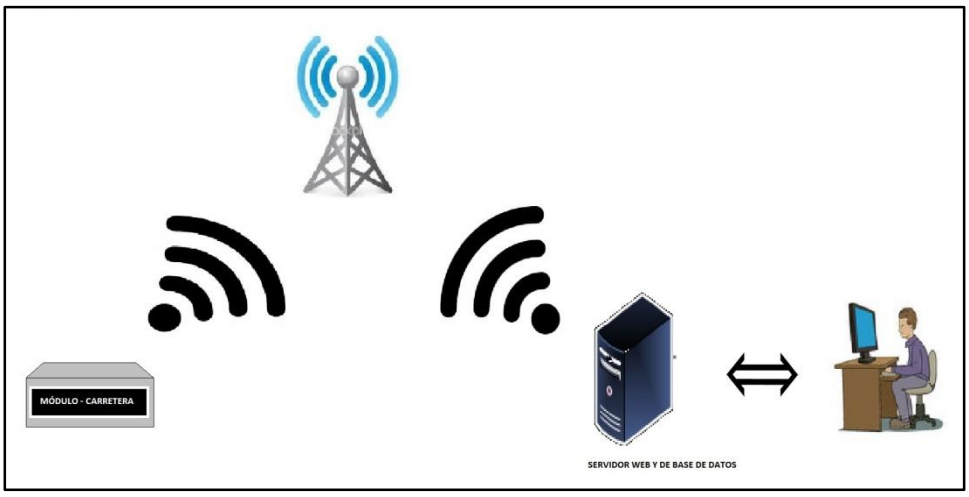

Figura 3.1. Esquema de la implementación

De aquí en adelante se explicará en detalle la implementación de cada una de las partes que lo conforma, empezando por el servidor hasta explicar detenidamente cada una de las partes constitutivas del módulo ubicado en la carretera.

3.4.1 Análisis y Diseño del portal web

Para el desarrollo de página web se ha utilizado el servidor virtual WampServer 3.0 para plataforma Windows, el cual permite gestionar los sitios web y todos los componentes del mismo.

El lenguaje de programación que se ha utilizado es PHP para el desarrollo de la página web. PHP es un lenguaje de programación de código abierto, y ofrece una gran variedad de herramientas para la elaboración del sitio web, cabe recalcar que se ha utilizado la versión PHP 7.0, y además se encuentra desarrollado en un sistema operativo Windows.

Para la interacción entre la base de datos Mysql y el servidor web se ha utilizado Ajax, debido a que trabaja de una manera asíncrona, permite realizar cambios en el portal sin interferir con la visualización y el comportamiento de la página. Se encuentra creada la librería "*foo.php*" la cual permite que el detalle de las carreteras sea insertado correctamente a la base de datos. En el *Anexo A* se puede encontrar un manual de usuario de la página web donde se describen cada una de sus funcionalidades.

```
if(isset($_GET['estado']))
{
          if(isset($_GET['idCarretera'])){
          cEstadoCarretera($_GET['idCarretera'],$_GET['estado']);
   }
   else{
          echo "llego estado pero no llego idCarretera";
   }
}
else{
   echo "no llego estado";
}

function cEstadoCarretera( $idCarretera, $estado){

//Conexion a la base de datos
$conexion3 =mysqli_connect("localhost","root","","carreteras");

$update= "UPDATE `carretera` SET `estado`='". $estado ."' WHERE
`id`= " . $idCarretera;

if (mysqli_query($conexion3, $update)){
     echo "<script>alert('Actualizada
Correctamente..!!')</script>";
}
else{
```

```
   echo "<script>alert('Error: " . $update . "<br>" .
mysqli_error($conexion3) . "')</script>";
}
}
```

3.4.2 Diseño de la base de datos

Para la creación y administración de la base de datos relacional se ha utilizado phpMyAdmin, la aplicación cliente por defecto de WampServer.

La base de datos consta básicamente de cuatro tablas, las cuales se detallan a continuación:

- **Usuario:** En esta tabla se almacena información general del sistema, tales como sus nombres, apellidos, cédula y edad; además de su correo el cual será utilizado para enviar las alertas al administrador y por último, un campo llamado prioridad el cual me permite distinguir entre los tipos de usuario (Administrador y Técnico) para limitar las acciones que pueden realizar.
- **Carretera:** Esta tabla contiene información general de la carretera, como su nombre, longitud, provincia, el número de carriles y el estado de la misma, el cual se actualiza desde la página web.
- **Suceso:** Es la tabla que almacena datos sobre los percances acontecidos en las carreteras. Cada registro consta de la fecha y hora de ocurrencia, el identificador de la carretera en la que sucedió, un campo que almacena qué tanto está obstruida la vía y la prioridad del suceso.
- **Cámara:** En esta tabla se guarda información de las cámaras instaladas en las carreteras para monitorearlas. Consta de un código, el identificador de la carretera donde fue instalada y una dirección para acceder a ella.

A continuación, se presenta el diagrama que muestra las relaciones entre las tablas de la base de datos:

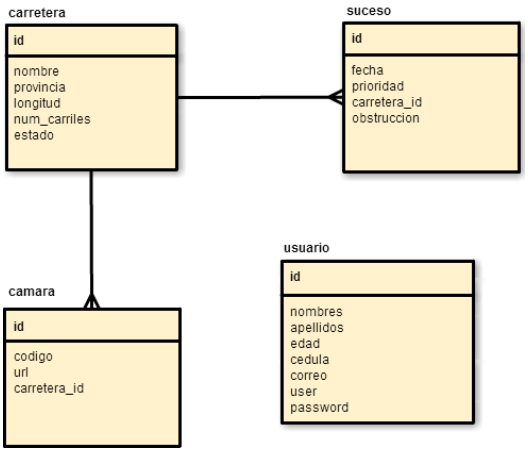

Figura 3.2. Diagrama Entidad-Relación.

3.4.3 Gráficas con librería HighCharts en PHP

En la página web también se incluye la opción Historial, en la cual el Administrador o los Técnicos podrán visualizar un listado de las carreteras que están siendo monitoreadas y escoger cualquiera para desplegar en la parte inferior una tabla con el listado de los sucesos ocurridos. Pero para una mejor visualización de los datos se incluye también dos enlaces que sirven para generar gráficas estadísticas de los sucesos en una nueva pestaña.

Para ello, usamos una librería de JavaScript llamada HighCharts (versión 4.1.5), la cual permite generar todo tipo de gráficas estadísticas ideales para aplicaciones web; tales como gráficos de barras, de pastel, de líneas, entre otras. Para implementar esta parte de la página se tenía a disposición algunas librerías, pero esta ofrece mayor variedad de gráficos, funciona con la mayoría de navegadores modernos, es robusta, sencilla y la posibilidad de poder exportar los gráficos generadas a distintos formatos como lo son: .pdf, .png o .jpg; además de brindar a los usuarios la opción de imprimirlos directamente desde el navegador.

Cabe recalcar que los gráficos generados son dinámicos ya que se generan con datos obtenidos mediante una consulta a la base de datos;

ya que todos los archivos con extensión. php que generan los gráficos inician con una conexión a la base de datos y su respectiva consulta.

A continuación, se muestra una imagen de una de las gráficas que se podrá generar en la página:

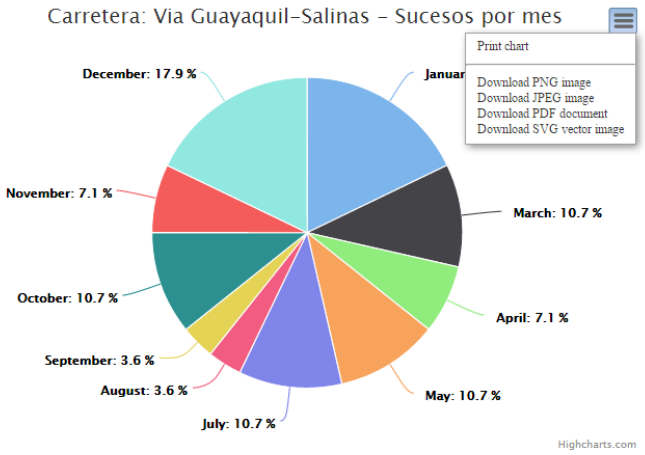

Figura 3.3. Gráfica de pastel generada con HighCharts.

Como se mencionó anteriormente, todos los gráficos generados cuentan con un menú desplegable en el cual aparecen las opciones para imprimir el gráfico y exportarlos a archivos con distintas extensiones.

3.4.4 Implementación del módulo en la carretera

Para la implementación del módulo en la carretera se utiliza la plataforma electrónica *Arduino Uno R3* la cual se encarga de tomar los datos leídos por lo sensores y procesarlos de tal manera que permita clasificar los acontecimientos en la carretera por prioridad; para posteriormente enviar las respectivas alertas.

El prototipo se basa en el uso de los Sensores Ultrasónicos Hc-sr04 conectados a la placa *Arduino Uno* y ubicados en pares a lo largo de la carretera, uno frente a otro; de tal manera que tomando las distancias medidas por los sensores y comparándolas con el ancho de la carretera

más la separación existente entre cada sensor y el borde de la misma, nos permitirá tener una medida de qué tanto está siendo obstruida la vía.
La comunicación entre el módulo ubicado en la carretera y el servidor que aloja la base de datos se realiza de manera inalámbrica utilizando el módulo WIFI ESP8266; por tanto, mediante este medio será enviada la información que se aloja en la base de datos.
Por tanto, a la placa Arduino estarán conectados los sensores ultrasónicos, una pantalla LCD 16x2 donde se muestran mensajes del estado de la vía; y por último estará conectado el módulo ESP8266 que se encarga del envío de la información.

3.4.5 Librería NewPing.h

Para la medición de las distancias utilizando los sensores ultrasónicos descargamos e importamos la librería NewPing.h. Con esta librería inicializamos dos objetos del tipo NewPing (uno por cada sensor), para poder manipularlos y medir las distancias de los obstáculos.
Para instanciar los objetos hacemos uso del constructor presentado a continuación:

```
NewPing sonar(trigger_pin, echo_pin [, max_cm_distance]);
```

En el constructor enviamos como parámetro los pines del Arduino que estemos utilizando como Trigger y Echo para cada sensor ultrasónico. Además, se puede especificar la máxima distancia (en cm) a la que puede detectar un obstáculo en la carretera, por defecto se inicializa con 500 cm.
Otro de los métodos que utilizamos es aquel que nos permite sensar las distancias de los obstáculos:

```
sonar.ping();
```

Este método envía un ping y retorna el tiempo en microsegundos que le toma al hecho llegar al sensor. Como el método presentado anteriormente

nos retorna un valor de tiempo, dicho valor retornado es dividido para una constante definida en la librería, `US_ROUNDTRIP_CM`.

3.4.6 Librería SoftwareSerial.h

Una de las limitantes encontradas al utilizar la placa Arduino Uno es que contamos con tan solo dos pines (0 y 1) para comunicación serial; lo cual representa un problema cuando debemos conectar módulos a la placa, en este caso debíamos conectar el módulo WIFI ESP8266. Para ello creamos una instancia de SoftwareSerial para comunicarnos con el módulo (justo antes del método Setup), utilizando el siguiente constructor:

```
SoftwareSerial mySerial (rxPin, txPin);
```

En el constructor debemos especificar cuáles son los pines de la placa Arduino que usaremos para enviar y recibir datos. Una vez que instanciamos el objeto utilizamos el método que nos permite definir la velocidad (en baudios) para la transmisión de datos. El método es el siguiente:

```
begin(speed);
```

3.4.7 Librería LiquidCrystal.h

Para informar a las personas sobre el estado de una vía, mostramos mensajes en una pantalla LCD, la cual es conectada al Arduino y controlada mediante el uso de la librería LiquidCrystal.h. Para ello instanciamos un objeto del tipo LiquidCrystal con el constructor:

```
LiquidCrystal(rs, enable, d4, d5, d6, d7);
```

En él debemos definir los pines del *Arduino* que vamos a utilizar para habilitar la pantalla LCD y pasar la información que se va a mostrar en ella. Al igual que con otras librerías debemos inicializar el objeto con el método ***begin(nrows, ncols)***; en el cual se debe especificar el número de

filas y columnas que tiene la pantalla, en este caso, la pantalla tiene dos filas de dieciséis columnas cada una.

Para enviarle el mensaje que debe mostrar hacemos uso del método ***write(data)***; al cual le enviamos como parámetro una cadena de caracteres que representa el mensaje a mostrar.

3.4.8 Plugin ESP8266 para Arduino IDE

El módulo ESP8266 es capaz de proporcionar otras funcionalidades aparte de servir para agregarle una conexión inalámbrica al *Arduino*; ya que internamente posee su propio procesador (con un poco menos de capacidad y potencia) que permite emular algunas de las funcionalidades que podríamos hacer con él.

Para ello utilizamos un plugin que ha sido desarrollado con el fin de que podamos programar el módulo directamente desde el *Arduino IDE*. De igual manera, tomando como base la librería WIFI de *Arduino*, se desarrolló una nueva, la cual se maneja de la misma manera que la oficial. Por tanto, comparando los costos entre el módulo ESP8266 y una WIFI Shield, y dependiendo de las funcionalidades del proyecto que se esté implementando, podría resultar mucho más conveniente trabajar con el módulo descrito anteriormente.

Existen dos formas de realizar esta parte, la primera sería descargar un IDE que ya trae configurado el plugin o podemos agregar el plugin a nuestro IDE principal.

Se ha optado por la segunda opción para lo cual es necesario que se tenga instalada la versión 4.6.1 del IDE de Arduino dado que el plugin que estamos agregando creará definiciones de nuevas placas y las incorporará al entorno de desarrollo; situación que no es permitida en versiones más antiguas. Para ello en la opción de "*Preferencias*" del menú archivo, se debe especificar la URL desde la cual se obtienen las placas adicionales que deseamos instalar. Una vez hecho esto, debe ingresar al Administrador de Tarjetas y buscar la opción correspondiente al módulo e instalar. Con ello, al momento de escoger la placa a la cual se desea

cargar el programa, se puede escoger la opción Módulo ESSP8266 genérico, como se muestra en la Figura 3.4:

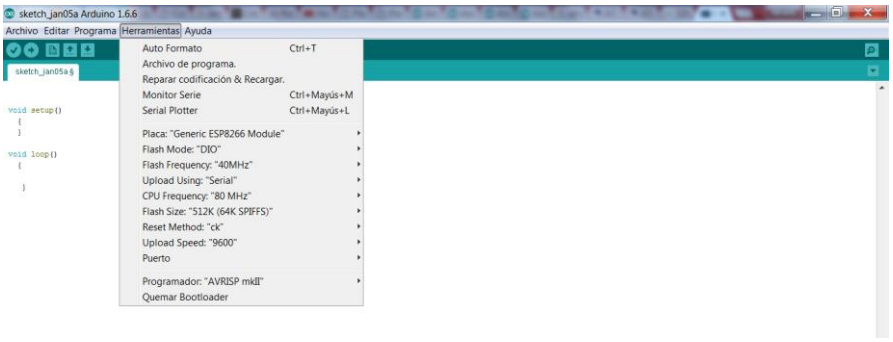

Figura 3.4. Instalación del plugin ESP8266 en el Arduino IDE.

Es importante tener en cuenta que una vez que escojamos la placa ESP8266, debemos configurar correctamente las características antes de iniciar la carga del programa, tales como velocidad de carga, frecuencia del CPU, entre otras.

3.4.9 Actualización del firmware del ESP8266

Para poder cargar un programa a nuestro módulo utilizando el Arduino IDE, primero se le actualiza el firmware. Para ello descargamos una versión más reciente del firmware (en nuestro caso descargamos la versión 0.9.5) y una pequeña aplicación que nos permite llevar a cabo el proceso. Para realizar la actualización, se debe conectar el firmware en modo de “flasheo” utilizando un conversor de USB a Serial tal como se muestra en la Figura 3.5:

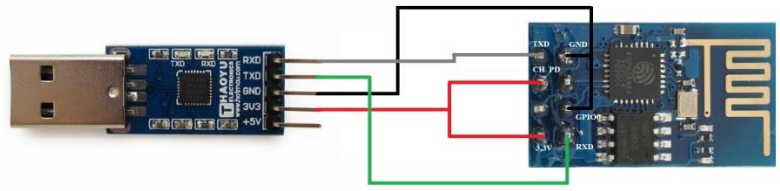

Figura 3.5. Conexión del Módulo ESP8266.

3.4.10 Código cargado al módulo ESP8266

Una vez que preparamos el módulo, procedemos a cargarle el código desarrollado para él. Este se basa en el uso de la librería *ESP8266WiFi.h*. En el código se definen dos variables constantes de tipo char que almacenan el SSID y contraseña de la red inalámbrica a la cual queremos conectarnos. Dicha conexión se la realiza utilizando el siguiente método:

```
Wifi.begin("ssid","password");
```

Luego, podemos verificar si la conexión se realizó con éxito haciendo uso del método ***Wifi.status()***.

El objetivo principal de este script es permitirle al módulo actuar como un pequeño servidor web, el cual estará esperando recibir peticiones por parte de algún cliente. Esto se realiza mediante el uso de objetos *WifiServer* y *WifiClient*.

Al inicio del script, inicializamos el objeto *WifiServer* especificando el número de puerto que será utilizado, esto lo realizamos de la siguiente manera:

```
WiFiServer server(80);
WiFiClient client;
```

Sabiendo que el método Setup se ejecuta una sola vez, en él empezamos a invocar los métodos que permiten al módulo conectarse a una red inalámbrica. A continuación se ejecuta repetidamente el método Loop(), en el cual se empieza validando la existencia de algún cliente que esté realizando alguna petición al módulo de la siguiente manera:

```
client = server.available();
```

En caso de que exista algún cliente realizando una petición, usamos el siguiente método que nos permite capturar los parámetros que fueron enviados:

```
String req = client.readStringUntil('\r');
```

Dependiendo del valor recibido imprimiremos por comunicación serial un carácter que el Arduino lo validará para mostrar el mensaje en la pantalla LCD, indicando que la carretera cambió de estado a abierta o cerrada.
Hasta ahora hemos descrito la parte del código en donde el módulo ESP8266 actúa como servidor, sin embargo, también puede cumplir el rol de cliente. Esto ocurre en el momento en que se guardan los datos leídos de los sensores, el módulo recibe por comunicación serial los parámetros relacionados con el suceso, para luego reenviarlos al realizar una petición al servidor que es quien se encarga de almacenarlos en la base. A continuación, se presenta la porción del código donde el módulo recibe los parámetros y los envía al servidor:

```
while (Serial.available()) {
      // get the new byte:
      char inChar = (char)Serial.read();
      // add it to the inputString:
      inputString += inChar;
      // if the incoming character is a newline, set a flag
      // so the main loop can do something about it:
      if (inChar == '\n') {
        stringComplete = true;
      }
  }
    String st="";
    String req="";
    // print the string when a newline arrives:
    if (stringComplete) {
      int l = inputString.length();
      st = inputString.substring(0,l-2);
       req = "GET /guardar_suceso.php?"+st+" HTTP/1.1";

      if(client.connect(http_site, http_port)){
        client.println(req);
        client.print("Host: ");
        client.println(http_site);
        client.println("Connection: close");
        client.println();
```

```
}
```

En las variables http_site y http_port se han almacenado la dirección IP del servidor de base de datos y el puerto utilizado para la comunicación.
El código puede ser visualizado de manera completa revisando el *Anexo B*.

3.4.11 Algoritmo de detección y almacenamiento de sucesos

Como hemos mencionado anteriormente, para la detección de obstáculos en las carreteras, utilizaremos sensores ultrasónicos colocados a ambos lados de la vía uno frente al otro. Estos sensores proporcionan la distancia a la que se encuentra un obstáculo, en caso de haberlo.
Los sensores toman una primera lectura de las distancias hasta algún obstáculo la cual se almacena en un par de variables; luego de esto viene un tiempo de espera antes de tomar una nueva lectura de ambas distancias, es decir manejamos cuatro variables que almacenarán distancias.
Luego de esto se realizan una serie de validaciones para asegurarnos de que efectivamente se trate de un obstáculo en la vía y no de algún vehículo circulando por ella o cualquier objeto colocado junto al sensor. Para estas validaciones se toman en cuenta el ancho de la carretera y la distancia de los sensores a la misma. La primera validación que se realiza permite reconocer si se trata del mismo objeto al validar que las dos parejas de distancias leídas sean las mismas. Luego de ello se procede a verificar que la obstrucción se encuentre sobre la carretera más no a un costado de ella. Y finalmente se clasifica el suceso de acuerdo a qué tanto esté obstruyendo la vía, al cual se le asigna un valor de prioridad. Finalmente, estas lecturas se convierten en parámetros que se envían al servidor para que se han almacenados en la base de datos. El código lo puede observar en mejor detalle en el *Anexo C*.

3.4.12 Envío de SMS y correo como alertas

Para alertar al administrador del sistema sobre algún acontecimiento en las vías se utilizan dos tipos de alerta: un mensaje de texto (SMS) y un correo, la cual para realizar los envíos se desarrollaron scripts en php.

Para enviar el correo electrónico primeramente se configura en el servidor la cuenta de Gmail que hará las veces de remitente de todos los correos enviados. Luego de esto se procede a desarrollar el script *SendMail.php* para realizar el envío automático del correo.

En la primera parte del código se realizan consultas a la base para extraer información de la carretera en donde se produjo el acontecimiento la cual se usa en el cuerpo del correo; además se realiza una consulta para obtener el correo del administrador. Una vez que se tengan los datos se utilizará la siguiente función:

```
mail($to, $subject, $mensaje)
```

Como vemos, se debe especificar como parámetros los siguientes datos:

- El correo de destino, en este caso el correo del administrador.
- Se debe especificar el asunto del correo.
- Por último, se detalla el cuerpo del mensaje.

El otro tipo de alerta utilizada es el envío de un SMS para el cual se usa el *API Sinch SMS*. Para esto se empieza creando una cuenta en Sinch y creamos una nueva App para que se generen automáticamente un par de códigos secretos necesarios para nuestro script *sendSMS.php*. Para obtener el número del administrador que debemos usar como destino del mensaje se realiza una consulta a la base de datos y armamos el *String* correspondiente al mensaje de la misma manera que en el caso del envío del correo.

Los códigos completos para el envío de las alertas se almacenan en los archivos llamados *SendMail.php* y *sendSMS.php*; los cuales se encuentra en el *Anexo D*.

3.4.13 Sistema de video vigilancia con Raspberry Pi 2

Finalmente se implementa un sistema de video vigilancia utilizando la Raspberry Pi 2 con una imagen de Raspbian (con versión de Kernel 3.18), la cual será almacenada en una tarjeta microSD de 4 GB. Tenemos adicionalmente que instalar el software Motion, que es un servicio de detección de movimiento; que permite ver en tiempo real lo que sucede en las carreteras.

Una vez instalado el software necesario, se procede a editar el archivo *motion.conf*; y el cambio más importante será deshabilitar la restricción de solamente permitir conexiones localmente. De esta manera se puede acceder al video en vivo simplemente realizando una petición ingresando esta dirección en el navegador:

```
http://ip-raspberry:puerto
```

En la página se puede habilitar en el menú la opción “Cámaras”, en la cual se muestra el listado de las mismas y la carretera en la que están ubicadas; además de un enlace que abrirá otra pestaña en la cual se podrá visualizar en vivo qué es lo que está sucediendo en las carreteras.

CAPÍTULO 4

4. RESULTADOS

Una vez que se realizó toda la implementación, se procedió a realizar las pruebas necesarias para comprobar el correcto funcionamiento del prototipo. Para ello se realizó pruebas para los tres tipos de prioridades como se va a presentar a continuación. Como era de esperarse, los obstáculos son detectados de manera inmediata por los sensores ultrasónicos y se envía los datos para guardarlos en la base de datos. Para comprobar que los sensores detectan correctamente los obstáculos, se imprime por puerto serial las lecturas de las distancias de cada uno de ellos hasta el obstáculo, además de la prioridad del suceso.

4.1 Suceso de alta prioridad

Un suceso es clasificado como de prioridad alta si logra obstaculizar más allá de la mitad de la carretera. Por tanto, al colocar el obstáculo entre los sensores, se pudo observar que en Monitor Serial del *Arduino IDE* se imprimió la prioridad como Alta y las distancias como se muestra a continuación:

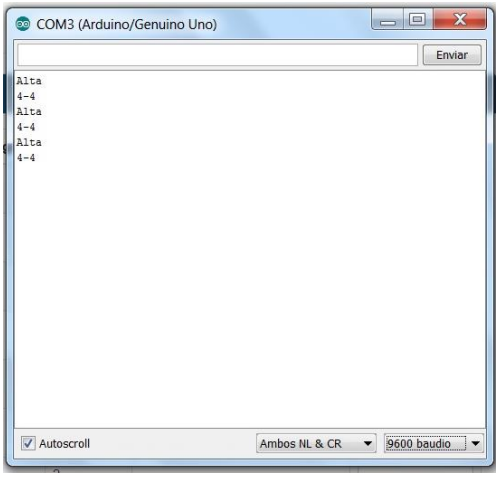

Figura 4.1. Detección de suceso de prioridad alta.

Como se puede observar en figura 4.1, las distancias de cada sensor al obstáculo son de 4 cm; y la distancia entre los sensores es de 30 cm, podemos decir que

el obstáculo es de prioridad alta. Luego procedemos a comprobar que el suceso fue almacenado en la base de datos, como se muestra en la figura 4.2:

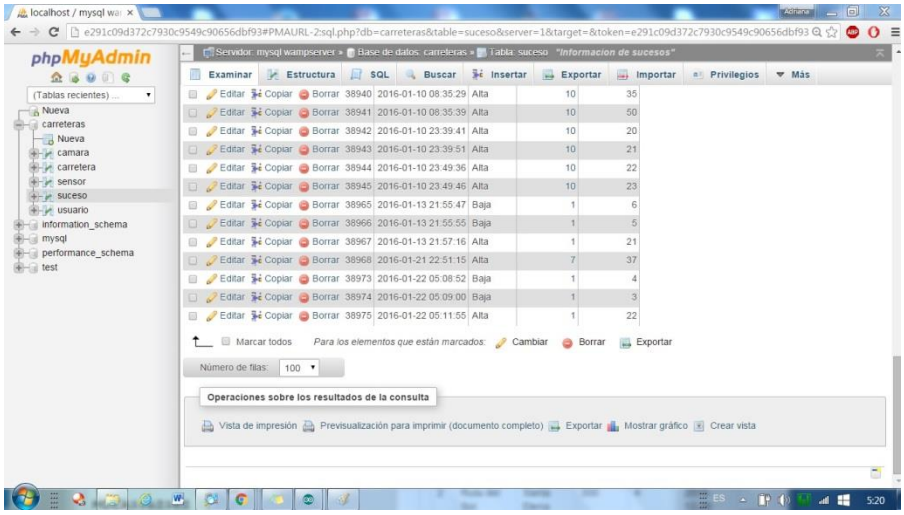

Figura 4.2. Almacenamiento de suceso de prioridad alta en la base.

Inmediatamente después de guardar el suceso se realiza el envío automático de las alertas por correo y mensaje de texto como se muestra en la figura 4.3. La rapidez con que se envían las alertas puede ser verificada comparando la hora de llegada de las mismas con la hora en que se realizó el registro en la base.

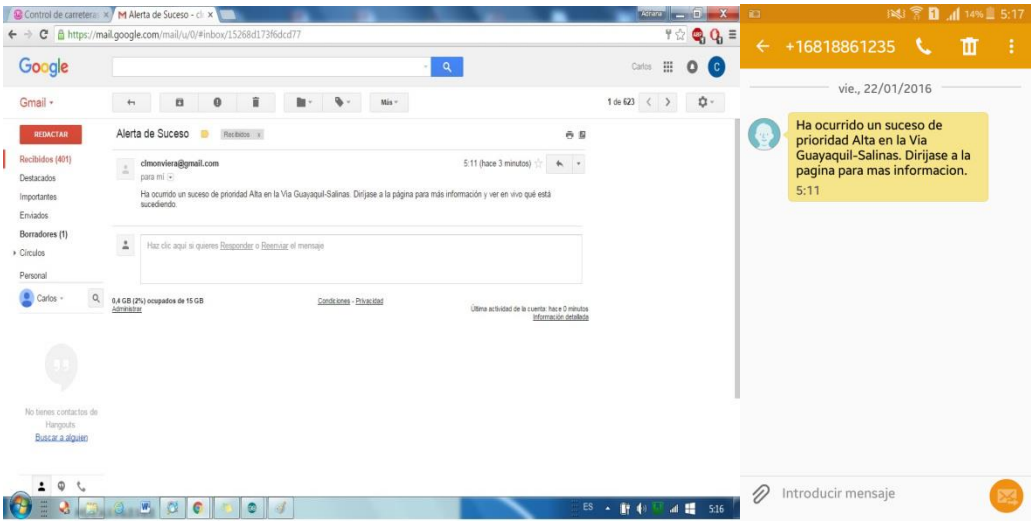

Figura 4.3. Alertas enviadas ante un suceso de prioridad alta.

4.2 Suceso de Prioridad Media

Un suceso es clasificado como de prioridad media si logra obstaculizar entre un cuarto y la mitad de la carretera. Por tanto, al colocar el obstáculo entre los sensores, se pudo observar que en el Monitor Serial del *Arduino IDE* se imprimió la prioridad como Media y las distancias como se muestra a continuación:

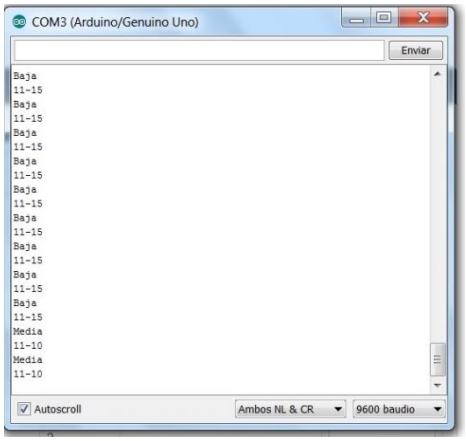

Figura 4.4. Detección de suceso de prioridad media.

Como podemos notar dado que las distancias de cada sensor al obstáculo son de 11 y 10 cm; y la distancia entre los sensores es de 30 cm, se puede decir que el obstáculo es de prioridad media. Luego procedemos a comprobar que el suceso fue almacenado en la base de datos, como se muestra en la figura 4.5:

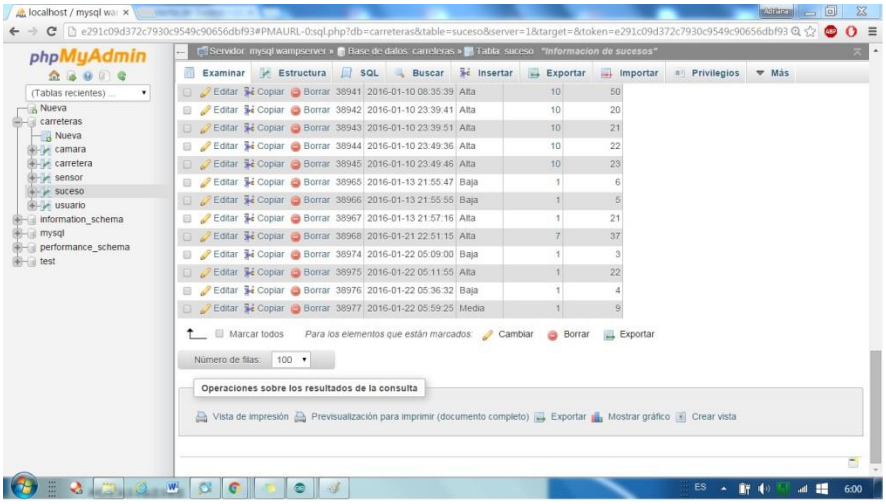

Figura 4.5. Almacenamiento de suceso de prioridad media en la base.

Al igual que el caso anterior, las alertas fueron enviadas de manera inmediata como se muestra en la figura 4.6:

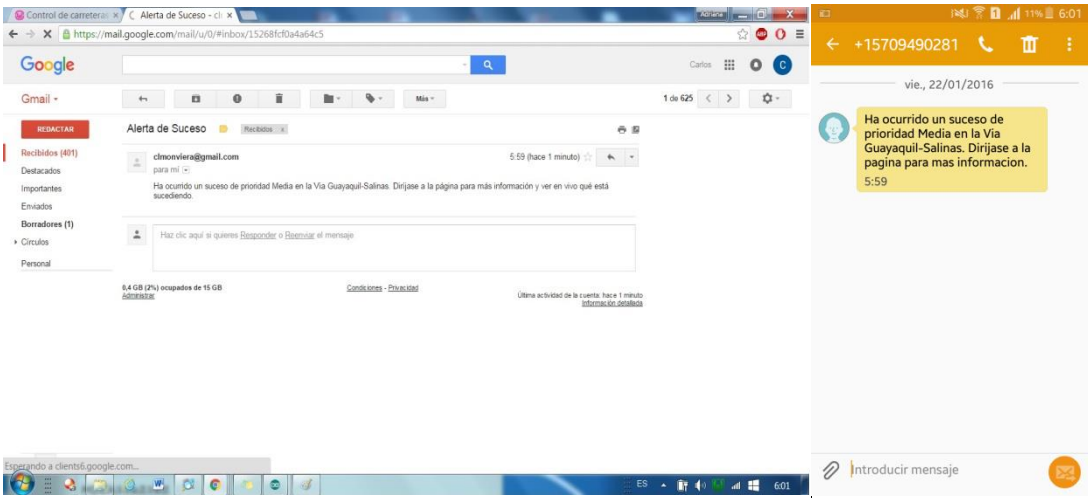

Figura 4.6. Alertas enviadas ante un suceso de prioridad media.

4.3 Suceso de Prioridad baja

Un suceso es clasificado como de prioridad baja si obstaculiza menos de la cuarta parte de la carretera. Por tanto, al colocar el obstáculo entre los sensores, se observa que en el Monitor Serial del *Arduino IDE* se imprimió la prioridad como Baja y las distancias como se muestra a continuación:

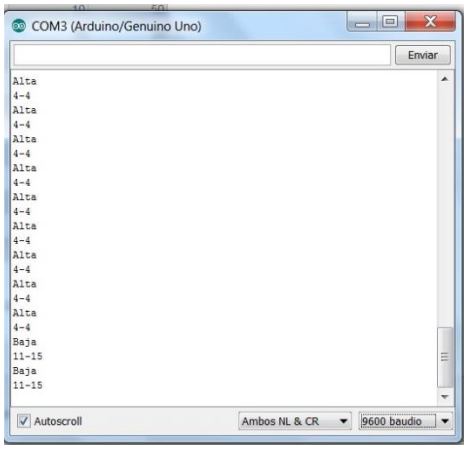

Figura 4.7. Detección de suceso de prioridad baja.

Podemos notar que las distancias de cada sensor al obstáculo son de 11 y 15 cm; y la distancia entre los sensores es de 30 cm, se puede decir que el obstáculo es de prioridad baja. Luego se comprueba que el suceso fue almacenado en la base de datos, como se muestra en la figura 4.8:

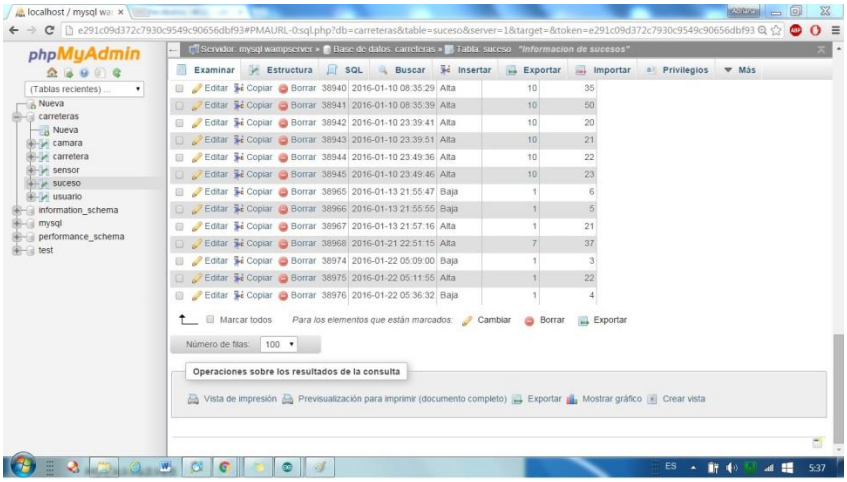

Figura 4.8. Almacenamiento de suceso de prioridad baja en la base.

El envío de las alertas se realizó de manera exitosa e inmediata como se puede apreciar en la figura 4.9:

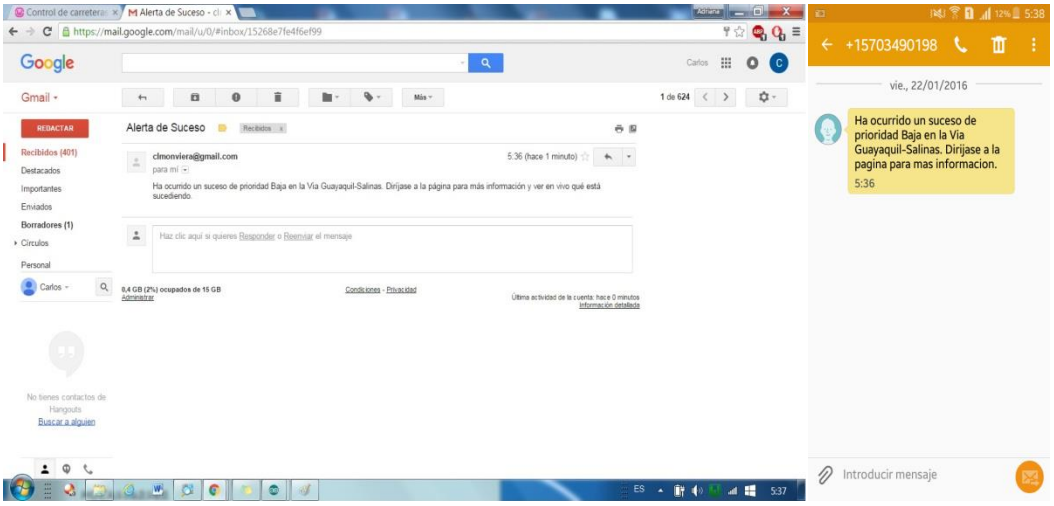

Figura 4.9. Alertas enviadas ante un suceso de prioridad baja.

4.4 Cierre y apertura de vías

Cuando el sistema es energizado y empieza a funcionar, automáticamente se coloca el mensaje de “Vía abierta” en la pantalla LCD, como se muestra en la imagen:

Figura 4.10. Mensaje mostrado para la apertura de la vía.

Luego, para cada uno de los sucesos se procedió a realizar el cierre de la vía desde la página web, donde al actualizar el estado de la carretera se muestra un mensaje de confirmación, ratificando el cambio realizado.

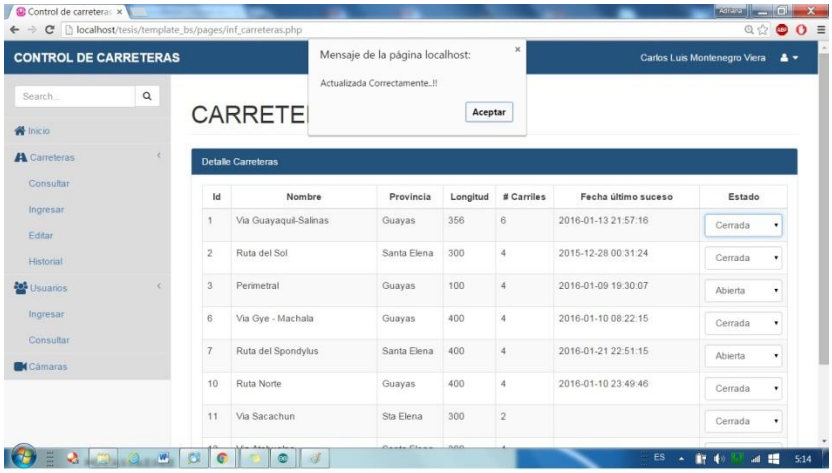

Figura 4.11. Cambio del estado de la vía.

El cambio del mensaje en la pantalla LCD tiene un retardo de alrededor de 3 segundos, pero finalmente el mensaje cambia a “Vía Cerrada”, actualizándose también en la base de datos.

Figura 4.12. Mensaje mostrado para el cierre de la vía.

CONCLUSIONES Y RECOMENDACIONES

CONCLUSIONES

1. El algoritmo para la detección de obstáculos, implementado en base a los sensores ultrasónicos, detecta y clasifica de manera eficaz los acontecimientos; además de asignarle la prioridad tomando en cuenta las distancias hasta los obstáculos encontrados.
2. El módulo WIFI ESP8266 utilizado para agregarle una conexión inalámbrica al módulo puede reemplazar sin ningún problema a la WIFI Shield de Arduino; ya que cumplió efectivamente con el objetivo de enviar los datos desde el módulo ubicado en la carretera hasta el servidor, sin mencionar la ventaja que representa su bajo costo.
3. La implementación de un sistema de video vigilancia en las carreteras le permitió a los usuarios del sistema tener un panorama más claro de qué es lo que está sucediendo en ellas, por lo que pueden tomar una mejor decisión al momento de cerrar una vía.

RECOMENDACIONES

1. Para implementar este prototipo es importante leer detenidamente las especificaciones de los módulos utilizados, ya que se da el caso que no todos trabajan con el mismo nivel de voltaje o corriente lo que podría conllevar a que los elementos puedan quemarse o simplemente no puedan siquiera encender por el bajo nivel de corriente que reciben.
2. Tener en cuenta el rango de distancias en el que trabaja el sensor ultrasónico utilizado en el prototipo para asegurarnos que no nos entregue lecturas incorrectas que puedan conllevar a almacenar datos erróneos en la base.

GLOSARIO DE TÉRMINOS

APSD: Son las siglas de **Automatic Power Save Delivery** (Ahorro de energía automático de entrega).

VoIP: son las siglas de **Voice over Internet Protocol** (Voz sobre Protocolo de Internet o Telefonía IP), una categoría de hardware y software que permite a la gente utilizar Internet como medio de transmisión de llamadas telefónicas, enviando datos de voz en paquetes usando el IP en lugar de los circuitos de transmisión telefónicos.

GSM: Son las siglas **Global System for Mobile communications** (Sistema Global para Comunicaciones Móviles), es un estándar desarrollado por el European Telecommunications Standards Institute (ETSI) para describir los protocolos de segunda generación (2G)digitales redes celulares utilizados por los teléfonos móviles.

GPIO: Del inglés **General Purpose Input/Output**, (Entrada/Salida de Propósito General) es un pin genérico en un chip, cuyo comportamiento (incluyendo si es un pin de entrada o salida) se puede controlar (programar) por el usuario en tiempo de ejecución.

PCB: Del inglés **Printed Circuit Board,** (circuito impreso) es la superficie constituida por caminos, pistas o buses de material conductor laminadas sobre una base no conductora. El circuito impreso se utiliza para conectar eléctricamente a través de las pistas conductoras, y sostener mecánicamente, por medio de la base, un conjunto de componentes electrónicos.

BIBLIOGRAFÍA

[1] A.C. Juan, “Sistema de Control y Monitoreo Vehicular utilizando tecnología RFID y envio de alertas mediante mensaje de texto,” Tesis de Grado, Fac. Ing.en Sist. Elect. e Ind. Univ. Tec. De Ambato, Ambato, Ecuador, 2015.

[2] A. Saburido, (2014, Mayo 20). 10 razones para usar arduino [Online]. Disponible en: http://www.modulo0tutoriales.com/10-razones-para-usar-arduino/

[3] N. Díaz, (2012, Julio 29). Arduino uno R3 características [Online]. Disponible en: http://www.taringa.net/post/hazlo-tu-mismo/15302861/Arduino-Uno-R3-caracteristicas-y-projecto-1-blink.html

[4] Anónimo, (2012). Módulo WiFi - ESP8266 [Online]. Disponible en: https://www.sparkfun.com/products/13678

[5] K. Soria, (2013, Septiembre 17). HC-SR04 sensor Ultrasónico [Online]. Disponible en: http://bkargado.blogspot.com/2013/09/todosobrehc-sr04.html

[6] R. Bourdon, (2005). WampServer [online]. Disponible en: http://www.wampserver.com/en/

[7] T. Páez, P. Gómez (2008, septiembre 7). Mysql Características [Online]. Disponible en: http://sistemaspyt.blogspot.com/2008/09/principales-caracteristicas.html

[8] Apache Software Foundation, (2011). Apache [Online]. Disponible en: https://es.opensuse.org/Apache

[9] O. Guzmán, (2015, Diciembre 8). Servidor Apache [Online]. Disponible en: https://prezi.com/zg2vtfpdbje4/servidor-apache/

[10] P. Roncancio, (2013, Septiembre 25), PHP [Online]. Disponible en: https://prezi.com/7prpgcwp849x/que-es-php/

[11] Anónimo, (2015, Diciembre 2), Sistema global para las comunicaciones móviles [Online]. Disponible en: https://es.wikipedia.org/wiki/Sistema_global_para_las_comunicaciones_m%C3%B3viles

[12] Earl, (Octubre 10). HC-SR04 [Online]. Disponible en: http://microcontrollerelectronics.com/distance-sensing/

[13] Anónimo, (2015, Noviembre 26), MYSQL [online]. Disponible en: https://es.wikipedia.org/wiki/MySQL

ANEXOS

Anexo A.

En este anexo se presenta una descripción de todas las opciones disponibles en la página web; detallando cómo y para qué puede utilizarse cada una de ellas.

El sitio web está conformado por una ventana de inicio de sesión en la cual el usuario deberá autenticarse para realizar el debido ingreso al sistema, cabe recalcar que esta opción solo podrá ser ejecutada por el administrador del sistema y los técnicos.

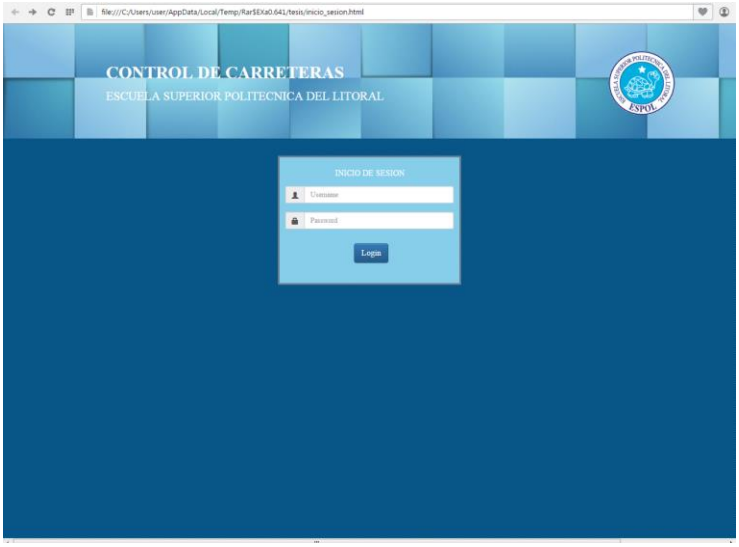

Figura A1. Pantalla de inicio de sesión.

Posteriormente aparece una página de bienvenida en la cual ya se puede apreciar el menú de opciones, el cual permite administrar los usuarios y las carreteras que intervienen en el desarrollo de este proyecto. Cabe recalcar que el número de opciones disponibles en el menú dependerá del rol que tenga la persona que ha iniciado sesión, ya que las opciones para administrar los usuarios del sistema solamente pueden ser vistas por el Administrador. A continuación se presenta una imagen donde se presentan la pantalla de inicio para un administrador y para un técnico:

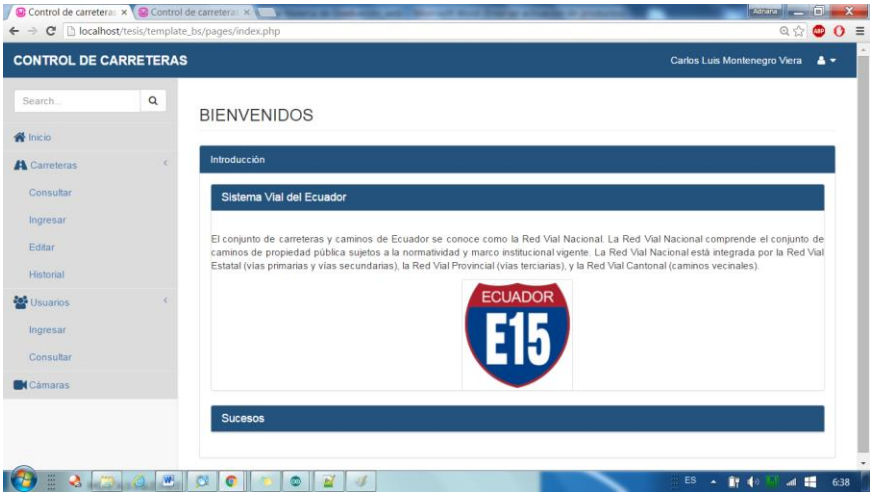

Figura A2. Pantalla de inicio del administrador.

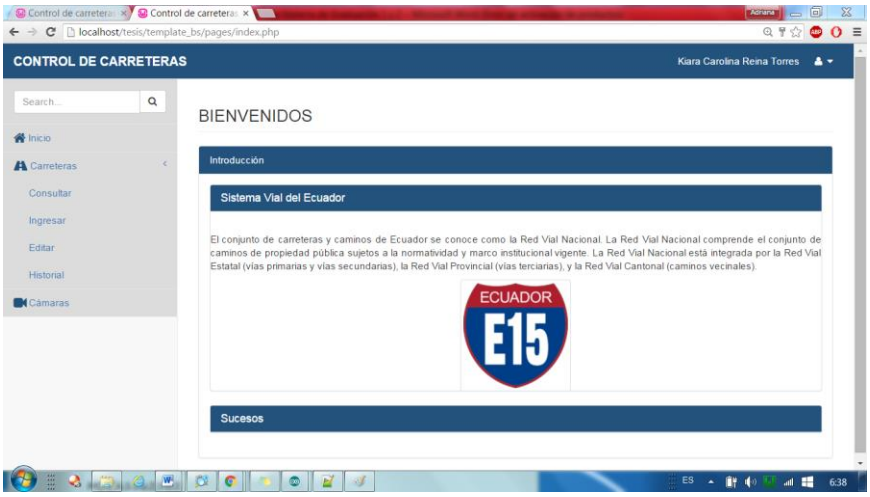

Figura A3. Pantalla de inicio del técnico.

En la sección de Carreteras, se encuentran cuatro opciones. La primera de ellas es la de Consultar, en la cual los usuarios podrán ver información general de las mismas y realizar el cierre o apertura cuando deseen.

Figura A4. Pantalla de consultas de carreteras.

La segunda opción es la de Ingresar, en la cual se pide ingresar todos la información requerida para registrar alguna nueva carretera que vaya a ser monitoreada.

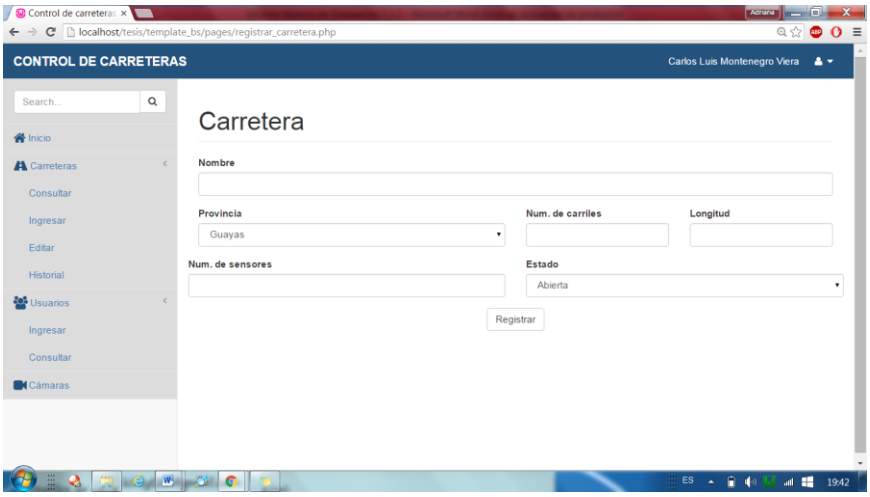

Figura A5. Pantalla de ingreso de carretera.

La siguiente opción es la de Editar, en la cual se podrá escoger alguna de las carreteras de una tabla pulsando el botón Editar, con lo que inmediatamente después de la tabla aparecerá la información actual de ella, la cual puede ser modificada y actualizada.

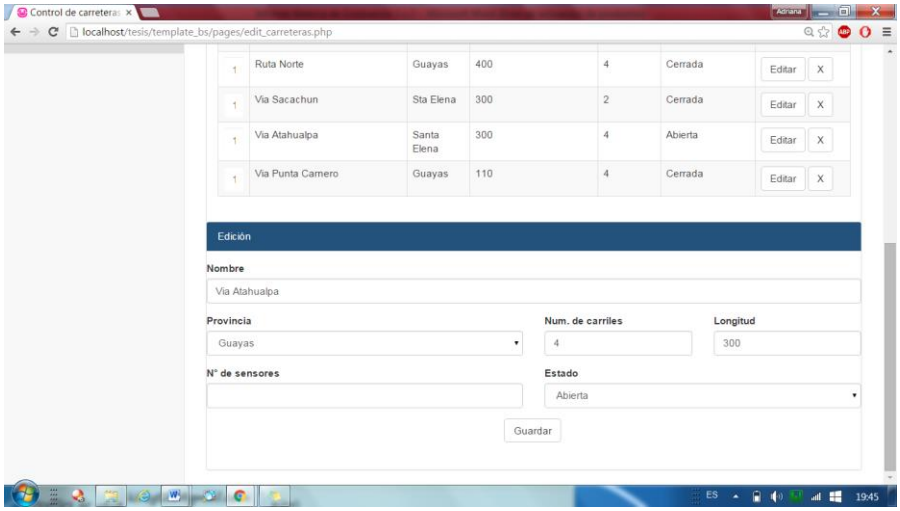

Figura A7. Pantalla de edición de carreteras.

La última opción es Historial, en la cual se presentan las carreteras en una tabla en donde, al presionar el botón Mostrar se despliega una tabla con el historial de sucesos en la carretera, además de dos enlaces para generar la gráficas estadísticas que se mencionaron anteriormente.

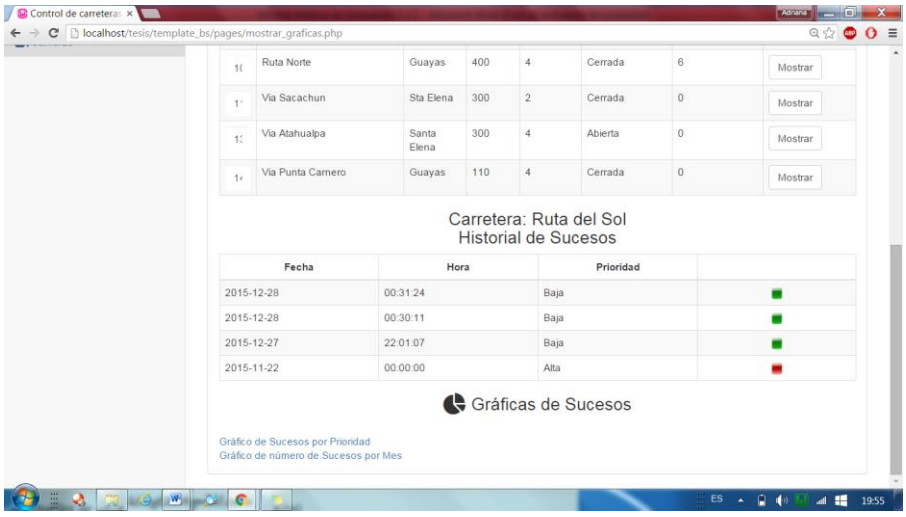

Figura A8. Pantalla de historial por carretera.

Luego aparecen las opciones de Usuarios. La primera opción es Ingresar, en la cual se puede crear un nuevo usuario del sistema, cabe recalcar que los datos deben ser ingresados correctamente debido a las validaciones que existen.

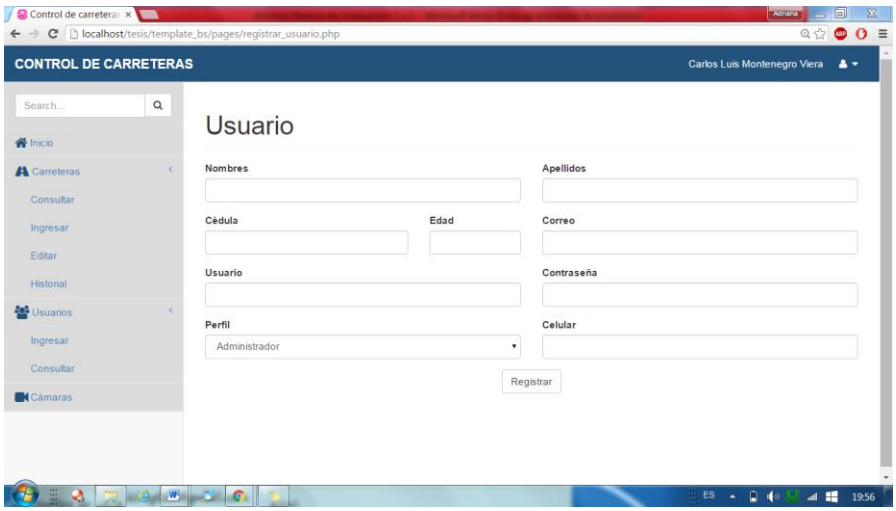

Figura A9. Pantalla de ingreso de usuario.

La otra opción corresponde a Consultar, en donde se puede observar todos los usuarios que pueden acceder al sistema, especificando su información general así como el rol que cumplen.

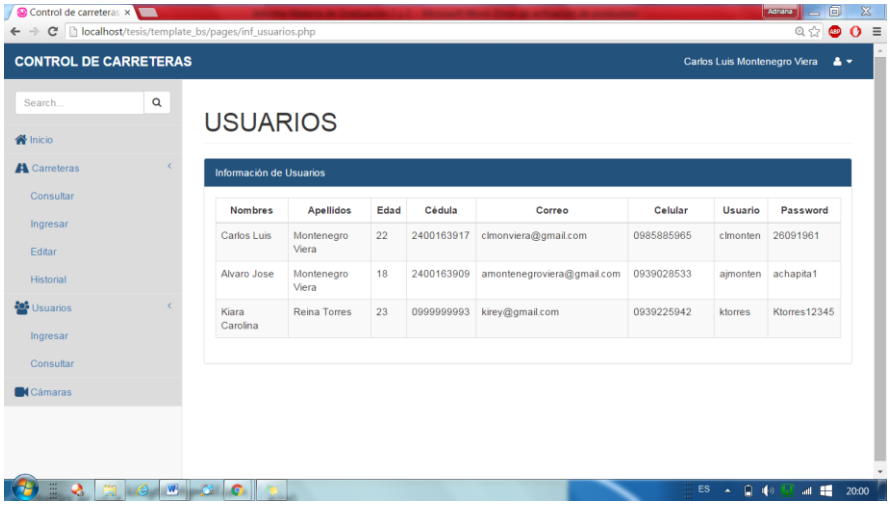

Figura A10. Pantalla de consulta de usuarios.

Por último se encuentra la opción Cámaras, que es donde los usuarios encontrarán una tabla con todas las cámaras instaladas y carretera donde están instaladas. Cada registro cuenta con un enlace que abre inmediatamente otra pestaña donde se puede visualizar lo que ocurre en cierto punto de alguna de las carreteras.

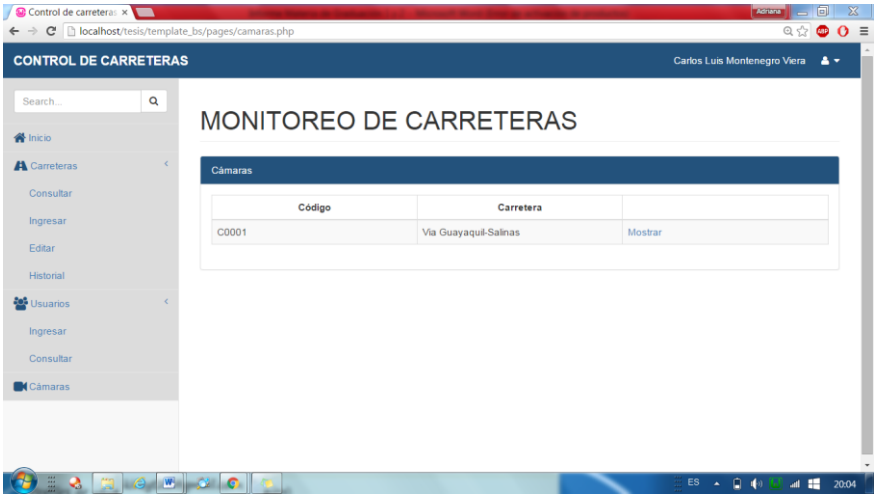

Figura A11. Pantalla monitoreo de las carreteras.

Anexo B.

En el presente Anexo se presenta el código que fue cargado en el módulo WIFI ESP8266.

```
#include <ESP8266WiFi.h>
#include <WiFiClient.h>


String inputString = "";         // a string to hold incoming data
boolean stringComplete = false;  // whether the string is complete

const char* ssid = "Montenegro_Viera";
const char* password = "prueba123";

// Remote site information
const char http_site[] = "192.168.0.102";
const int http_port = 80;

// Create an instance of the server
// specify the port to listen on as an argument
WiFiServer server(80);
WiFiClient client;

void setup() {
  Serial.begin(9600);
  delay(3000);
  // reserve 200 bytes for the inputString:
  inputString.reserve(200);
  // Connect to WiFi network

  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
  }

  // Start the server
  server.begin();
}

void loop() {
  // Check if a client has connected
  client = server.available();
  if (client) {

  // Wait until the client sends some data
  while(!client.available()){
    delay(1);
  }

  // Read the first line of the request
  String req = client.readStringUntil('\r');
  client.flush();

  // Match the request
  if (req.indexOf("/gpio/4") != -1){
    Serial.println("4");
```

```
  }
  else if (req.indexOf("/gpio/5") != -1)
    Serial.println("5");
  else if (req.indexOf("/gpio/6") != -1)
    Serial.print("6");
  else if (req.indexOf("/gpio/7") != -1)
    Serial.print("7");
  else {
    client.stop();
    return;
  }
  }
  client.flush();

  while (Serial.available()) {
      // get the new byte:
      char inChar = (char)Serial.read();
      // add it to the inputString:
      inputString += inChar;
      // if the incoming character is a newline, set a flag
      // so the main loop can do something about it:
      if (inChar == '\n') {
        stringComplete = true;
      }
  }
    String st="";
    String req="";
    // print the string when a newline arrives:
    if (stringComplete) {
      int l = inputString.length();
      st = inputString.substring(0,l-2);
      if(st.indexOf("obs") != -1){
        req = "GET /guardar_suceso.php?"+st+" HTTP/1.1";
      }else{
        req = "GET /sendmail.php?"+st+" HTTP/1.1";
      }

      if(client.connect(http_site, http_port)){
        client.println(req);
        client.print("Host: ");
        client.println(http_site);
        client.println("Connection: close");
        client.println();
      }
    delay(5000);
    // clear the string:
    inputString = "";
    stringComplete = false;
    }
  client.flush();
}
```

Anexo C.

En este Anexo se presenta el código cargado en la placa Arduino Uno R3, el ejecuta el algoritmo para detectar obstáculos mediante los sensores ultrasónicos; además de modificar los mensajes que se muestran en la pantalla LCD cuando recibe la orden del módulo WIFI por comunicación serial.

```
#include <NewPing.h>
#include <SoftwareSerial.h>
#include <String.h>
#include <stdlib.h>
#include <LiquidCrystal.h>

#define S1_TRIGGER_PIN  4
#define S1_ECHO_PIN     5
#define S2_TRIGGER_PIN  6
#define S2_ECHO_PIN     9
#define MAX_DISTANCE 300

NewPing sonar1(S1_TRIGGER_PIN, S1_ECHO_PIN, MAX_DISTANCE);
NewPing sonar2(S2_TRIGGER_PIN, S2_ECHO_PIN, MAX_DISTANCE);
SoftwareSerial mySerial(7, 8);
LiquidCrystal lcd(10, 11, 12, 13 , 14, 15);
SoftwareSerial WIFI(3, 2); // RX | TX

void setup() {
  mySerial.begin(19200);               // the GPRS baud rate
  Serial.begin(9600);
  WIFI.begin(9600);
  delay(10000);
  lcd.begin(16,2);
  lcd.setCursor(0,1);
  lcd.write("Via abierta");
  delay(60000);
}

void loop() {
 if(WIFI.available()){
    char c = WIFI.read();
    if(c=='6'){
      lcd.clear();
      lcd.setCursor(0,1);
      lcd.write("Via cerrada");
    }
    if(c=='7'){
      lcd.clear();
      lcd.setCursor(0,1);
      lcd.write("Via abierta");
    }
 }

  const int SEPARACION_CARRETERA=5;
  const int ANCHO_CARRETERA=20;

  int prioridad=-1;
```

```
  int dist1 = (sonar1.ping_median() / US_ROUNDTRIP_CM);
  int dist2 = (sonar2.ping_median() / US_ROUNDTRIP_CM);

  delay(5000);

  int d1tmp = (sonar1.ping_median() / US_ROUNDTRIP_CM);
  int d2tmp = (sonar2.ping_median() / US_ROUNDTRIP_CM);

  if(dist1==d1tmp & dist2==d2tmp){
    if(dist1<(ANCHO_CARRETERA+SEPARACION_CARRETERA) &
dist2<(ANCHO_CARRETERA+SEPARACION_CARRETERA)){
      if((dist1+dist2) >=
((ANCHO_CARRETERA+(2*SEPARACION_CARRETERA))*3/4)){
        prioridad=0;
        Serial.println("Baja");
        Serial.print(dist1);Serial.print("-");Serial.println(dist2);
      }else if(((dist1+dist2) <
((ANCHO_CARRETERA+(2*SEPARACION_CARRETERA))*3/4)) & ((dist1+dist2) >=
((ANCHO_CARRETERA+(2*SEPARACION_CARRETERA))/2))){
        prioridad=1;
        Serial.println("Media");
        Serial.print(dist1);Serial.print("-");Serial.println(dist2);
      }else if((dist1+dist2) <
((ANCHO_CARRETERA+(2*SEPARACION_CARRETERA))/2)){
        prioridad=2;
        Serial.println("Alta");
        Serial.print(dist1);Serial.print("-");Serial.println(dist2);
      }
    }
    int obstruccion = (ANCHO_CARRETERA + (2*SEPARACION_CARRETERA)) -
(dist1 + dist2);
    guardarSuceso(prioridad, obstruccion);

  }
  //delay(15000);
}

void guardarSuceso(int prioridad, int obstruccion)
{
  String params = "calle=1&pr=";
  params += String(prioridad);
  params += "&obs=";
  params += String(obstruccion);

  WIFI.println(params);
  delay(3000);
}
```

Anexo D.

En el presente Anexo se presenta el código de los archivos SendMail.php y sendSMS.php, utilizados para realizar el envío de un correo y un mensaje de texto respectivamente.

SendMail.php.

```
<?php
$conexion=mysqli_connect("localhost","root","","carreteras");
$carretera = mysqli_real_escape_string($conexion, $_GET['calle']);
$prioridad = mysqli_real_escape_string($conexion, $_GET['pr']);
$p="";

$busqueda1 = "select C.nombre from carretera C where
C.id=$carretera;";
$resultado1=mysqli_query($conexion, $busqueda1);
$fila1=mysqli_fetch_array($resultado1);
$nombre = $fila1[0];

if ($prioridad == "0") {
    $p="Baja";
} elseif ($prioridad == "1") {
    $p="Media";
} else {
    $p="Alta";
}

$mensaje = 'Ha ocurrido un suceso de prioridad ' . $p . ' en la ' .
$nombre . '. Diríjase a la página para más información y ver en vivo
qué está sucediendo.';

$busqueda2 = "select U.correo from usuario U where
U.prioridad='Administrador';";
$resultado2=mysqli_query($conexion, $busqueda2);
$fila2=mysqli_fetch_array($resultado2);

$to = $fila2[0];

$subject = 'Alerta de Suceso';


if(mail($to, $subject, $mensaje))
       echo "Correo enviado";
else
       echo "Falla en el envio";

?>
```

sendSMS.php

```
<?php
$conexion=mysqli_connect("localhost","root","","carreteras");
$carretera = mysqli_real_escape_string($conexion, $_GET['calle']);
```

```
$prioridad = mysqli_real_escape_string($conexion, $_GET['pr']);
$p="";

$busqueda1 = "select C.nombre from carretera C where
C.id=$carretera;";
$resultado1=mysqli_query($conexion, $busqueda1);
$fila1=mysqli_fetch_array($resultado1);
$nombre = $fila1[0];

if ($prioridad == "0") {
    $p="Baja";
} elseif ($prioridad == "1") {
    $p="Media";
} else {
    $p="Alta";
}

$mensaje = 'Ha ocurrido un suceso de prioridad ' . $p . ' en la ' .
$nombre . '. Dirijase a la pagina para mas informacion.';

$busqueda2 = "select U.celular from usuario U where
U.prioridad='Administrador';";
$resultado2=mysqli_query($conexion, $busqueda2);
$fila2=mysqli_fetch_array($resultado2);

$numero = $fila2[0];
$n_sin_cero = substr($numero, 1);


$key = "d1f269d5-5f4e-47da-87e6-fbb921fca04b";
$secret = "32B1L4Eh5E+cZepzMQc7FQ==";
$phone_number = '+593' . $n_sin_cero;
$user = "application\\" . $key . ":" . $secret;
$message = array("message"=>$mensaje);
$data = json_encode($message);
$ch = curl_init('https://messagingapi.sinch.com/v1/sms/' .
$phone_number);
curl_setopt($ch, CURLOPT_POST, true);
curl_setopt($ch, CURLOPT_USERPWD,$user);
curl_setopt($ch, CURLOPT_POSTFIELDS, $data);
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, false);
curl_setopt($ch, CURLOPT_HTTPHEADER, array('Content-Type:
application/json'));
$result = curl_exec($ch);
if(curl_errno($ch)) {
    echo 'Curl error: ' . curl_error($ch);
} else {
    echo $result;
}
curl_close($ch);
?>
```