UNIVERSIDAD TECNOLÓGICA ISRAEL

TRABAJO DE TITULACIÓN EN OPCIÓN AL GRADO DE:

INGENIERO EN ELECTRÓNICA DIGITAL Y TELECOMUNICACIONES

TEMA: DESARROLLO DE UN PROTOTIPO PARA EL CONTROL Y MONITOREO AUTOMÁTICO DEL INGRESO Y SALIDA DE PASAJEROS EN UN BUS INTERPROVINCIAL CON ALERTAS SMS.

AUTOR: ROLANDO JAVIER OÑA OÑA

TUTOR: Mg. RENÉ ERNESTO CORTIJO LEYVA.

QUITO- ECUADOR

AÑO: 2019

DECLARACIÓN

Yo, Rolando Javier Oña Oña, declaro bajo juramento que el trabajo aquí descrito es de mi auditoria, que no ha sido previamente presentado para ningún grado o calificación profesional y que he consultado las referencias bibliográficas que se incluyen en este documento.

……

Rolando Javier Oña Oña

CERTIFICACIÓN DEL TUTOR

UNIVERSIDAD TECNOLÓGICA ISRAEL

APROBACIÓN DEL TUTOR

En mi calidad de tutor del trabajo de titulación certifico:

Que el trabajo de titulación **"DESARROLLO DE UN PROTOTIPO PARA EL CONTROL Y MONITOREO AUTOMÁTICO DEL INGRESO Y SALIDA DE PASAJEROS EN UN BUS INTERPROVINCIAL CON ALERTAS SMS.**", presentado por el Sr. Rolando Javier Oña Oña, estudiante de la carrera de Electrónica Digital y Telecomunicaciones, reúne los requisitos y méritos suficientes para ser sometido a la evaluación del Tribunal de Grado, que se designe, para su correspondiente estudio y calificación.

Quito D.M. Viernes 12 Julio del 2019

TUTOR

……..

Ing. René Ernesto Cortijo Leyva, Mg

AGRADECIMIENTOS

Este trabajo de titulación es el resultado de mi esfuerzo y dedicación, por esto agradezco a mi tutor de tesis, Ing. René Ernesto Cortijo Leyva Mg., que gracias a su conocimiento y sabiduría me encuentro culminando una de mis metas. Mis agradecimientos a mis profesores de esta prestigiosa universidad quienes fueron puntos claves de la información que necesitaba para el desarrollo de este trabajo de titulación.

DEDICATORIA

Este trabajo de titulación está dedicado con mucho afecto y amor a mis padres ya que gracias a ellos estoy culminando una de mis metas, con la ayuda de sus consejos y su apoyo incondicional en este camino hacia el éxito.

TABLA DE CONTENIDO

LISTA DE FIGURAS

LISTA DE TABLAS

RESUMEN

El desarrollo del prototipo para el control y monitoreo automático del ingreso y salida de pasajeros en un bus interprovincial con alertas SMS, tiene como propósito llevar un registro del ingreso y salida de pasajeros, llevar un monitoreo de pasajeros a través de la utilización de dispositivos o elementos electrónicos que ayuden a detectar dichos procesos, registrar el número de paradas realizadas, además de promover el uso obligatorio del cinturón de seguridad dentro de las unidades de transporte, logrando así reducir la cantidad de personas fallecidas durante algún siniestro o accidente que pueda ocurrir durante el viaje y por último dar a conocer cualquier eventualidad programada al usuario autorizado a través de un SMS.

Para desarrollar el prototipo se indagó diferentes soluciones que se encuentran vigentes en mercado de la electrónica actual, de la misma manera se consultó sobre el funcionamiento de los mismos y verificar si cumplen con las exigencias necesarias para desarrollar el objetivo principal planteado en este proyecto de titulación. El prototipo fue desarrollado en dos partes fundamentales: la parte electrónica y *software*. La parte electrónica está compuesta por el microcontrolador Arduino Mega 2560, finales de carrera, sensor magnético, módulo laser, módulo SIM808 GSM/GPS y un microcomputador (Raspberry PI). La parte de *software* se encuentra ligado a interfaz web y la base de datos que permite la visualización de datos en ese instante, además del envió de alertas SMS, el compilado de todos estos dispositivos electrónicos hace posible el desarrollo del prototipo.

El prototipo funciona de la siguiente manera: cuenta con un sensor magnético en la puerta principal de la unidad, en el cual al momento de ser cerrada o abierta este evento se registra dentro de una base de datos, adicionalmente cuenta de un módulo laser en la puerta del pasillo cuyo propósito es la de registrar ingreso y salida de pasajeros dentro de la base de datos, adicional a esto para el monitoreo de pasajeros se colocan finales de carrera en los cinturones de seguridad de cada asiento. Todos estos eventos serán notificados en ese instante a través de un SMS al usuario autorizado.

Palabras Claves: Automático, prototipo, LAMP, SMS, configuración.

ABSTRACT

The purpose of the development of the prototype for automatic control and monitoring of the entry and exit of passengers on an interprovincial bus with SMS alerts, is to keep a record of the entry and exit of passengers, monitor passengers through the use of devices or electronic elements that help detect such processes, record the number of stops made, in addition to promoting the mandatory use of seat belts within the transport units, thus reducing the number of people killed during a disaster or accident that may occur during the trip and finally to announce any eventuality programmed to the authorized user through an SMS.

To develop the prototype, we investigated different solutions that are current in the current electronics market, in the same way we consulted about their operation and verify if they meet the necessary requirements to develop the main objective proposed in this project. . The prototype was developed in two fundamental parts: the electronic part and software. The electronic part consists of the Arduino Mega 2560 microcontroller, limit switches, magnetic sensor, laser module, SIM808 GSM / GPS module and a microcomputer (Raspberry PI). The software part is linked to the web interface and the database that allows the visualization of data at that moment, in addition to the sending of SMS alerts, the compilation of all these electronic devices makes possible the development of the prototype.

The prototype works in the following way: it has a magnetic sensor in the main door of the unit, in which at the time of being closed or open this event is recorded in a database, additionally it has a laser module in the hallway door whose purpose is to register entry and exit of passengers within the database, in addition to this for the monitoring of passengers are placed limit switches in the seat belts of each seat. All these events will be notified at that moment through an SMS to the authorized user.

Keyworks. Automatic, prototype, LAMP, SMS, configuration.

Descripción de los capítulos

Dentro del primer capítulo, se especifica la información adecuada a los dispositivos electrónicos utilizados en el desarrollo del prototipo como son: microcontrolador Arduino Mega, módulo SIM808 GSM/GPS, sensor magnético, finales de carrera, módulo laser, etc.

En el segundo capítulo se describe acerca de la metodología de investigación usada para el desarrollo del proyecto de titulación.

En el tercer capítulo se describe la propuesta a ser presentada para el desarrollar el presente proyecto de titulación.

Dentro del cuarto capítulo se detalla los pasos que se siguió para el desarrollo del prototipo y también se describen pruebas de funcionamiento.

En el último capítulo se describen las conclusiones en base a los objetivos plateados y las recomendaciones que se necesita para realizar el desarrollo del prototipo, adicional a esto se detallar las fuentes bibliográficas consultadas con sus respectivos anexos.

INTRODUCCIÓN

Antecedentes.

En la actualidad llevar un control y monitoreo de pasajeros se lo hace de forma escrita y visual, en muchas ocasiones no refleja su veracidad dentro de las unidades de transporte, lo que conllevará a un proceso ineficiente e inseguro. Existen métodos para realizar un conteo de personas que suben a la unidad mediante barras magnéticas y cámaras de video, pero no presentan un registro del lugar en donde ingreso a la unidad o en donde se bajó de la unidad. Por otro lado, los sensores infrarrojos, sensores de presencia y sensores ultrasónicos ayudan detectar la presencia de objetos, animales o personas.

Los prototipos para dar seguimiento a los vehículos mediante GPS, son una opción, pero no abarcan registros de conteo de pasajeros ni mucho menos de llevar un control. En muchas ocasiones, estos prototipos pretenden registrar la hora de llegada, hora de salida y la ruta que se ha cumplido cuando el vehículo se encontró en movimiento.

El internet de las cosas (IoT) es el próximo paso en las redes de nueva generación en el cual las telecomunicaciones han ido evolucionando permitiendo controlar y monitorear a objetos en tiempo real con gran confiabilidad de los datos, mediante el uso de sensores y enlaces de red básica o robusta.

La importancia de este proyecto de titulación radica en el diseño y construcción de un prototipo para el control y monitoreo en tiempo real para pasajeros dentro de un bus de transporte interprovincial a través del uso de sensores y enlaces para la transmisión de la información, aumentando la eficiencia de los vehículos interprovinciales, disminuyendo la tasa de mortalidad dentro de los autobuses con el uso permanente del cinturón de seguridad y permitiendo así cumplir con una de las leyes de tránsito.

Planteamiento del problema.

La aglomeración de personas en las diferentes ciudades del Ecuador y la necesidad de trasportase de un lugar a otro, proporciono a que las compañías de transporte interprovincial en los últimos años deban abastecerse de más unidades para cumplir con la demanda de usuarios que se movilizan diariamente por las diferentes rutas que tiene el país.

En el Ecuador existen cerca de 6500 unidades de transporte interprovincial en el cual una o varias flotas cuenta con un socio o dueño, en la mayoría de las ocasiones los accionistas de las unidades de transporte contratan a choferes y ayudantes para operar dichos buses, por lo que el dueño no tiene una visualidad de lo que está pasando dentro de la unidad a menos que se comunique por teléfono con el chofer o ayudante que no es muy práctico. (FENACOTIP, 2018).

Por otro lado, la mayoría de las unidades de transporte interprovinciales modernos cuentan con cinturones de seguridad, pero no los usan, ya sea por la falta de desconocimiento o incomodidad al momento de viajar. En el Ecuador el pasado 11 de Diciembre del 2018 la disposición de la Agencia Nacional de Tránsito (ANT) ordenó que los buses interprovinciales que no tengan cinturones de seguridad no salgan de los terminales, por lo que las unidades que no dispongan de cinturones de seguridad deberán implementarlo de forma obligatoria. (El Comercio, 2018)

En la actualidad la mayoría de unidades de transporte no cuentan con un sistema para llevar un control y monitoreo de ingreso y salida de pasajeros de forma automática, ya sea este por costos demasiado altos o desconocimiento de los mismos y en muchas ocasiones se necesita de un alto conocimiento de electrónica y programación para poder implementar un sistema como este.

Otro punto clave, es el de poder controlar que los usuarios utilicen de forma obligatoria el cinturón de seguridad permitiendo así cumplir con lo dispuesto por la Agencia Nacional de Tránsito, y mantener al tanto al socio o dueño de la compañía de los eventos que están sucediendo en ese momento dentro de la unidad.

Justificación.

Muchas unidades de transporte de pasajeros no llevan un control de pasajeros de forma adecuada y en otras no hacen uso obligatorio del cinturón de seguridad por lo que el desarrollo de este proyecto radica en llevar un control del ingreso y salida de pasajeros, así como el monitoreo de los usuarios dentro de las unidades de transporte en el cual se observe el uso obligatorio del cinturón de seguridad.

Se dice que el uso del cinturón de seguridad reduce la posibilidad de muerte de hasta un 75% de pasajeros que viajan dentro de las unidades de transporte, por lo que este proyecto de titulación se busca impulsar el uso obligatorio del cinturón de seguridad para que de esta manera se pueda salvar vidas cuando ocurra un accidente tránsito.

Además, saber la cantidad de usuarios que suben y bajan de la unidad es indispensable para llevar un control diario o mensual según lo requiera la cooperativa o socio de la compañía, otro de los beneficios que se obtiene con el control adecuado de pasajeros es la de evitar que los usuarios viajen en los pasillos de la unidad de transporte.

Objetivos.

Objetivo general.

Desarrollar un prototipo para el control y monitoreo automático del ingreso y salida de pasajeros en un bus interprovincial con alertas SMS.

Objetivos específicos.

- Definir la cantidad y tipo de dispositivos electrónicos que van a formar parte del control de presencia de personas dentro del autobús y monitoreo automático del ingreso y salida de pasajeros.

- Desarrollar la programación respectiva para controlar y monitorear el ingreso y salida de usuarios, paradas realizadas mediante un registro, y enviar una notificación SMS cada vez que ocurra dichos eventos.

- Diseñar el diagrama de conexión y ubicación de los diferentes dispositivos electrónicos, módulo principal del prototipo necesarios para el control y monitoreo de transporte de pasajeros dentro del autobús.

- Realizar la comunicación entre Arduino y Raspberry a través del lenguaje de programación Python para leer los datos que se registran en el Arduino y almacenarlos en una base de datos.

- Crear una base de datos para el almacenamiento, interpretación y visualización de los mismos mediante el uso de herramientas como son: Servidor Apache, PHP, PhpMyAdmin y MySQL, dentro del módulo Raspberry.

- Implementar el sistema de control y monitoreo de transporte de pasajeros, en una maqueta que permita visualizar el funcionamiento del mismo.

- Realizar pruebas de funcionamiento del dispositivo, para detectar y corregir cualquier fallo de operación en el prototipo.

Alcance

El prototipo propuesto trabajará con sensores de fin de carrera ubicados en el interior de los cinturones de seguridad en cada butaca del autobús el cual permitirá tener un control y monitoreo de los mismos. Estos operaran cuando el conductor inicie la ruta o recorrido. Todos los finales de carrera llegaran a una placa de *hardware* libre en la cual se almacenarán dichos datos, además se dispondrá de un sensor magnético en la puerta principal para notificar cuando la misma sea accionada. Contará también de un transmisor y receptor laser el cual permitirá conocer si el pasajero está ingresando o saliendo de la unidad.

Los socios o dueños de la unidad podrán instalar los fines de carrera realizando una modificación al cinturón de seguridad, los demás elementos electrónicos que intervienen en el prototipo se los debe instalar con la ayuda o asesoramiento técnico en electrónica para de esta manera garantizar el funcionamiento de forma rápida y eficiente, se garantizará que los datos que lleguen al Arduino se podrán presentar en una interfaz gráfica amigable para conocer los eventos que han ocurrido durante el viaje.

Se realizará la programación respectiva para notificar al socio o dueño de la compañía de los eventos que ocurran dentro de la unidad, tales como monitoreo de butacas, ingreso y salida de pasajeros, apertura o cierre de la puerta principal, siempre y cuando el transmisor cuente con un plan de SMS para poder enviar mensajes.

Esto permitirá, que se haga uso obligatorio del cinturón de seguridad en cada una de las unidades de transportes interprovincial, además de que el socio o dueño tenga la transparencia de lo que está sucediendo en ese momento, y finalmente obtener un registro detallado de todo lo acontecido al finalizar el viaje para futuros análisis o referencias.

CAPÍTULO 1
FUNDAMENTACIÓN TEÓRICA

1.1 Antecedentes.

La necesidad de movilizarse de un logar a otro, ya sea desde lugares sumamente alejados o cortos se lo ha realizado desde la antigüedad, impulsando la economía y el desarrollo entre las ciudades. La aglomeración de personas en las diferentes ciudades y la necesidad de desplazarse de un lugar a otro fue unos de los factores primordiales para la formación de cooperativas de transporte que ofrecieran dicho servicio, los registros de pasajeros y conteo de los mismos se los llevaba de forma manual y visual, en muchas ocasiones los pasajeros viajaban en el pasillo de la unidad.

Con el paso de los años, se ha implementado varios tipos de sistemas para realizar el control de ingreso y salida de pasajeros como torniquetes móviles, el cual es colocado en la puerta cercana al pasillo de la unidad, en buses que cumplen rutas locales dentro de la ciudad es común ver barras electrónicas en la puerta de ingreso y salida de las unidades ya que estas fueron diseñadas para el uso exclusivo dentro de la ciudad y para este tipo de buses.

En el mercado electrónico se dispone de varios sistemas de monitoreo de pasajeros que ingresan y salen de la unidad, como son el de cámaras, torniquetes y barras electrónicas, pero ninguno de estos permite conocer que el usuario tenga abrochado el cinturón de seguridad o a su vez mantener informado que butaca o asiento se ha ocupado indicando el lugar, fecha y hora en el cual sucedió dicha eventualidad.

1.2 Sistemas de contador de pasajeros BUSAE.

Este sistema permite conocer el movimiento de los usuarios que ingresan y salen de la unidad a través de su sistema de visión estereoscópica, en el cual consiste de un par de cámaras ubicadas en las puertas de ingreso y salida de la unidad las mismas que detectan el

movimiento de las personas y que son rastreados a través de imágenes subsiguientes permitiendo así conocer si entra o sale de unidad, este tipo de sistemas se los usa en las ciudades urbanas donde existe gran aglomeración de personas que abordan los diferentes sistemas de transporte terrestre, es decir, son diseñados exclusivamente para este tipo de situaciones. (Busae, 2018).

En el mercado actual este sistema es muy costoso, ya que para obtener un reporte o monitoreo se debe cancelar una mensualidad, adicionalmente se debe colocar una cámara de seguridad el cual permita monitorear dicho sistema.

1.3 Sistema de control de ingreso y salida de pasajeros por torniquete.

El sistema de control de ingreso y salida de pasajeros por torniquete es un sistema mecánico y electrónico, en el cual se acciona tras verificar su autorización de forma manual, visual o electrónica cada vez que pasa una persona. Estos torniquetes cuentan con un contador electrónico el cual se acciona al pasar por el mismo.

Sus usos más comunes son en edificaciones, campus y en acceso a andenes de sistemas de transportes, este sistema no es muy conveniente usarlo dentro de autobuses ya que presenta una incomodidad y dificultad para pasar mujeres embarazadas y de la tercera edad. A continuación, se detalla las características que se tiene al usar un torniquete dentro de un autobús: (Siasa, 2018):

1.4 Características principales

- Difícil para pasar con bolsos, carteras, mochilas.
- No se puede pasar con niños en brazos
- Hay que empujar con fuerza y avanza de forma abrupta
- Material pesado frio y poco amigable.
- Puede ser evadido con facilidad

Figura 1.1 Torniquete

Fuente: **(Siasa, 2018)**

1.5 Barreras electrónicas contadoras de pasajeros.

Este sistema permite detecta cuando una persona ingresa o sale de la unidad, es decir, detecta el sentido en el transita, utilizan sensores opto acopladores para cumplir con su objetivo, este sistema no indica la posición actual en la que el pasajero abordo o abandono la unidad lo cual presenta una desventaja, pero a su vez un principio fundamental para la construcción del prototipo.

Es uno de los más simples y económicos que existen, su principio de funcionamiento se trata de poner un haz de luz en una zona de paso, cada vez que una persona atraviesa el haz, este se interrumpe el cual es interpretado como si una persona a ingresado o ha salido de la unidad de transporte según sea el caso.

1.6 Elementos necesarios para la construcción del prototipo.

Una vez analizado, las diferentes soluciones que se dispone en el mercado para realizar un control de ingreso y salida de pasajeros, en alguno de ellos presentan ciertas ventajas y desventajas, y en algunos casos no cumplen con los requerimientos necesarios, por tal motivo se procede a describir y mencionar los elementos o componentes necesarios para la

construcción del prototipo que permita controlar y monitorear el ingreso y salida de pasajeros, con alertas SMS.

1.6.1 Módulo SIM808 GSM/GPS

SIM808 es un módulo que combina dos funciones principales de comunicación tanto para envió y recepción de mensajes, realizar o recibir llamadas la cual se combina con la tecnología GPS para obtener la posición en latitud y longitud. Gracias a su pequeño consumo de energía este módulo puede ser conectado a baterías de litio que son de uso común en los teléfonos celulares, además de ser compatible con A-GPS el cual el módulo se lo puede controlar mediante comandos AT de ser necesario. Este dispositivo electrónico está conectado al Arduino Mega 2560 el cual enviará mensajes de altera (SMS) de los eventos que vayan a acontecer en ese momento. A continuación, se detalla las características principales del mismo. (Prometec, 2018)

- **Características principales**

  - Las bandas de frecuencia en la que opera son: (850/900/1800/1900) MHz
  - Compatible con la tecnología 2G
  - Se puede usar para realizar llamadas a través de manos libres
  - Capacidad para enviar y recibir mensajes SMS
  - Capacidad para enviar y recibir datos a través GPRS (TCP/IP, HTTP, FTP, etc).
  - Integra un receptor de GPS dentro del módulo SIM808
  - Compatible con comandos AT para la gestión del módulo GSM
  - Posee vibrador integrado para llamadas
  - Solo se puede insertar tarjeta SIM estándar 2G y se debe consultar la cobertura (preguntar cobertura con el proveedor)
  - El protocolo NMEA del GPS es compatible con tarjeta SIM estándar

Figura 1.2 Módulo SIM808 GSM/GPS

Fuente: (Prometec, 2018)

1.6.2 Arduino Mega 2560

El Arduino Mega 2560 está diseñado para proyectos más complejos que el Arduino Uno, además presenta las características apropiadas que se acoplan al prototipo que se desarrollará.

Se basa en el microcontrolador ATMEGA2560 posee una cantidad de 54 pines cuyo propósito son de entrada/salida (15 usados como PWM), cuenta con 16 pines analógicos, y cuatro puertos serial (UARTS), conector ISP, una conexión USB *jack* de alimentación, un cristal de 16 MHz, y botón de *reset*. (Tecmikro, 2018)

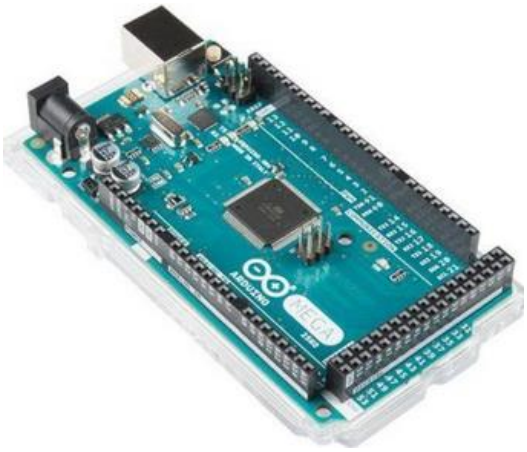


Figura 1.3 Arduino Mega 2560

Fuente: (Tecmikro, 2018)

- **Características Arduino Mega 2560.**

  - Se basa en el ATMEGA2560
  - Voltaje de alimentación recomendado: 7 a 12 volts.
  - Alimentación absoluta dentro de: 6 a 20 volts.
  - Entradas y salidas digitales 54
  - Entradas PWM 15 pines
  - Entradas analógicas 16 pines
  - Corriente de salida en los pines: 20 mA
  - Voltaje de 3.3 V con 50 mA
  - Memoria flash 256 KB para programas
  - Memoria RAM de 8KB
  - Memoria EEPROM 4KB
  - 16 MHz de reloj de frecuencia
  - Pin 13 de propósito general

1.6.3 Final de carrera.

Llamados también finales de carrera, son interruptores que divisan la posición de un dispositivo mediante accionamiento mecánico, este elemento electrónico ayudara a detectar si la persona está usando o no el cinturón de seguridad, ya que será acoplado dentro del mismo el cual se accionará cuando el usuario se haya abrochado el cinturón, de esta manera se podrá visualizar con exactitud que numero se asiento o butaca está ocupando el pasajero en ese momento, permitiendo así realizar un monitoreo del mismo. (Quiminet, 2018)

Figura 1.4 Final de carrera

Fuente: (Quiminet, 2018)

1.6.4 Diodo Laser

Este dispositivo electrónico tiene la capacidad de emitir un haz de luz el cual puede ser visible o no, en esta ocasión se usará el módulo KY-008 el cual tiene integrado por una cabeza laser de color rojo de 650 nm y una resistencia. Este módulo posee la ventaja de ajustar el grosor de puntero moviendo hacia la izquierda o derecha su calibrador, se recomienda no mirar directamente al laser ya que podría afectar o tener consecuencias en la visión. (Dualtronica, 2018)

Figura 1.5 Diodo laser

Fuente: (Dualtronica, 2018)

- **Características módulo KY-008**

  - Opera a 5 V
  - Su potencia de disipación es de 5 mW
  - Longitud de onda de 650 nm.
  - Corriente de 40 mA
  - Rango de temperatura es de 10° C a 40° C

1.6.5 Receptor Laser

Este dispositivo electrónico tiene la capacidad de detectar un haz de luz de un puntero laser ya que cuando la luz es enfocada en el sensor este cambia su valor de voltaje de salida, este sistema es usado en conteo de personas, detector de intrusos y transmisión de datos a distancia, este dispositivo no se recomienda usarlo con luz solar directa o entorno artificial ya que el módulo puede operar de forma incorrecta. (Vistronica, 2018)

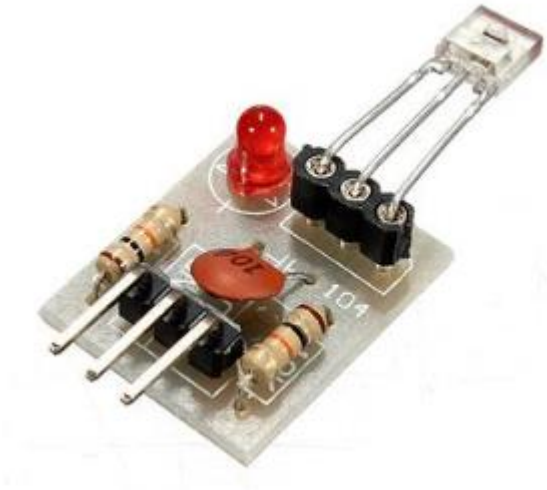


Figura 1.6 Receptor laser

Fuente: (Vistronica, 2018)

- **Características módulo receptor laser**

  - Opera a 5 V
  - Distancia efectiva entre emisor y receptor es de 0,1 m – 1,5 m
  - Rango de temperatura es de 10° C a 40° C

1.6.6 LCD 16x2.

Es un dispositivo que permite la visualización de información o contenido necesario para mostrar, el *display* de cristal líquido 16x2 está compuesto 2 filas de 16 caracteres y esta administrado por un microcontrolador que se encuentra integrado en el mismo, este

dispositivo electrónico es de mucha utilidad ya que se usa para desplegar la información de los asientos ocupados y libres que se disponga en dicho momento. (Ardumotive, 2018)

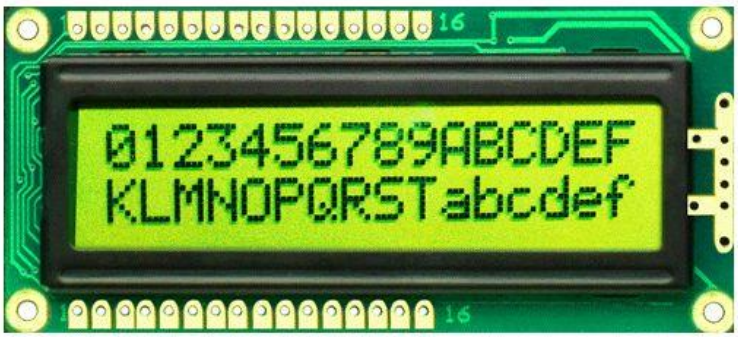


Figura 1.7 *Display* LCD 16x2

Fuente: (Ardumotive, 2018)

1.6.7 Raspberry PI.

Es un computador súper pequeño del tamaño de una tarjeta de crédito, de bajo costo y poco consumo de energía, generalmente en esta clase de ordenadores se ejecutan sistemas operativos basados en Linux u *Open Software*, además de un ser un ordenador súper pequeño Raspberry PI incorpora funciones de electrónica, con pines GPIO y de comunicación serial (*UART*), los cuales pueden ser empleados en proyectos de robótica, mecatrónica y electrónica. (Raspberry PI, 2018)

Raspberry PI trabaja bajo un sistema operativo que corre en Linux, es sistema operativo se llama *Raspbian Jessie*, su principal característica es que maneja una interfaz gráfica, que la hace amigable similar a la de Windows, siendo así muy fácil de usar.

Otras de las ventajas primordiales que se tiene al usar Raspberry es la de poder implementar un servidor como tal, en este prototipo se realizará una estructura LAMP por su acrónimo en inglés (Linux, Apache, MySQL/MariaDB, PHP/Perl/Python), donde se implementará una tecnología de desarrollo libre.

Dentro de Raspberry se almacenará el sitio web dentro la base de datos correspondiente la cual se podrá visualizar la información de todos los eventos, con el objetivo de que esta información esté disponible para el socio de la unidad o personal autorizado en cualquier momento.

Figura 1.8 Raspberry PI 3

Fuente: (Raspberry PI, 2018)

1.6.8 Servidor Web HTTP

Apache es un *software* de *web server* gratuito y de código abierto, su función principal es de establecer una conexión entre un servidor y los diferentes navegadores web que existen. Este servidor funciona bajo la plataforma de Linux y Windows, el cual resultará muy conveniente usarlo. (Hostinger, 2018)

- **Características del servidor Apache**

  - Soporte de seguridad SSL y TLS
  - Compatible con leguajes de programación como Perl, PHP, Phyton
  - Utiliza el protocolo HTTP
  - Es multiplataforma

1.6.9 PHP

Es un lenguaje de programación muy popular de código abierto, conveniente para realizar el desarrollo de la página web, en el que puede ser introducido en HTML lo que significa que dentro de un mismo archivo se podrá combinar código PHP y código HTML, de esta manera se conseguirá crear páginas web dinámicas que permitan mostrar la información en tiempo real. (PHP, 2018)

- **Ventajas de usar PHP**

  - Tiene compatibilidad con otros programas
  - Permite realizar una programación estructurada
  - La sintaxis de usar PHP es similar a C
  - PHP es Open Source
  - Permite conectar con una gran cantidad de motores de bases de datos

1.6.10 MySQL

Es gestor de bases de datos, el cual es usador principalmente para desarrollo de páginas web en conjunto con diferentes leguajes de programación como PHP o Java. (ItSoftware, 2018)

- **Características de MySQL**

  - Su velocidad y facilidad de uso
  - Su capacidad, es decir, que los clientes se conectan simultáneamente
  - Conectividad y seguridad
  - Es gratuito
  - De distribución abierta

1.6.11 Administrador de MySQL

PhpMyAdmin es un *software* que facilita administración y la gestión de la bases de datos creadas en dentro de MySQL, gracias a este programa se puede acceder desde un navegador de internet al gestor de bases de datos, este programa cuenta con la facilidad de poder modificar los datos almacenados, crear y eliminar usuarios de MySQL, importar y exportar bases de datos etc. (Ecured, 2018)

- **Características de PhpMyAdmin**

  - Posee una interfaz intuitiva
  - Soporta la mayoría de características de MySQL
  - Facilidad de importación de archivos CSV y SQL
  - Exporta archivos en formato: CSV, SQL, XML, PDF
  - De distribución abierta

1.6.12 Python

Es un lenguaje de programación interpretado, lo cual no hace falta la necesidad de un compilador como lo son otros lenguajes, además de su fácil utilización este lenguaje de programación ayudara la interconexión entre los datos que envié el Arduino Mega 2560 a la Raspberry PI, interpretándolos y enviándolos a la base de datos. (Geekelectronica, 2018)

- **Características de Python**

  - Lenguaje de alto nivel
  - Las variables de comprueba en tiempo de ejecución
  - No dispone de licencia para programar
  - Multiplataforma (Windows, Linux, MAC)
  - Se ejecuta sin necesidad de un compilador.
  - Facilidad de uso en proyectos de electrónica con Raspberry PI.

1.6.13 Herramienta de diseño para base de datos

SQL Workbench es un programa diseñado con el fin crear de bases de datos, muy intuitivo y fácil de usar, a través de este programa se desarrollará las tablas y relaciones que la base de datos necesitará para guardar datos e interpretándolos. (Ubunlog, 2018)

- **Características de SQL *Workbench***

  - Permite configurar parámetros de conexión
  - Proporciona la capacidad de ejecutar consultas SQL
  - Gran visualidad de tablas
  - Modelar diagramas entidad relación.

CAPÍTULO 2

MARCO METODOLÓGICO

2.1 Introducción.

En este proyecto se desenvolvió empleando dos tipos de indagación: aplicativo y exploratorio. Es de tipo aplicativo ya que se emplea la utilización y aplicación de los todos los conocimientos científicos y técnicos, adquiridos a lo largo del proceso de toda la formación universitaria y práctica, además la información o guía adquirida en el proceso formativo sobre la tecnología actual. Es exploratorio porque se va abordar acerca de un prototipo para un sistema de transporte interprovincial, en el mercado se encontró pocas soluciones similares, pero de distintas aplicaciones y necesidades.

2.2 Investigación exploratoria.

Se utilizó el método de investigación exploratoria, ya que la ejecución de este tipo de proyecto que se ha seleccionado no ha sido explorada ampliamente en las compañías de buses de transportes interprovincial, además que con este tipo de proyecto se podrá controlar el uso adecuado del cinturón de seguridad dentro de las unidades de transporte con la ayuda de módulos de electrónicos, comunicaciones, programación en microcontroladores, programación y desarrollo de páginas web, permitiendo así la visualización en tiempo real de los eventos que sucederán dentro la unidad para así obtener información más verídica.

2.3 Investigación bibliográfica-documental

El desarrollo de este proyecto de titulación se de investigación radica el uso de la modalidad bibliográfica, debido a que en primer lugar se revisó e investigó proyectos semejantes, que tengan alguna relación con el tema registrado, además de tener un enfoque más amplio como el seleccionado en este proyecto para su desarrollo.

Para esta investigación se apoyó primordialmente en la información recopilada de revistas internet, folletos y alguna otra información que trate sobre el tema propuesto, enfocándose en las técnicas y elementos electrónicos utilizados para el control y monitoreo de ingreso y salida de pasajeros.

2.4 Población y muestra.

En esta ocasión no fue necesario desarrollar muestras de población, muestras estadísticas, etc., debido a que el proyecto se enfoca mucho más en lo experimental y exploratorio.

CAPÍTULO 3

PROPUESTA

3.1 Descripción general del proyecto.

Para poder lograr realizar el desarrollo y construcción del prototipo este constará de modo general con los siguientes elementos:

Se dispondrá de interruptores finales de carrera en cada uno de los asientos en el cual se modificará cada uno de los cinturones de seguridad para colocar de la mejor manera posible el dispositivo electrónico donde cuyo propósito es el de enviar al Arduino una señal de que si el usuario está haciendo uso del cinturón de seguridad.

También estará constituido de módulos de transmisor y recepción laser, el cual identificará si el pasajero ingresa o abandona la unidad, procesando así una señal que llegará al Arduino y finalmente se dispondrá de un sensor magnético en la puerta principal del autobús el cual permitirá conocer el instante en que se abre o cierra la puerta.

El cinturón de seguridad que se dispone en cada uno de las butacas estará compuesto por un final de carrera, el cual funcionará como un interruptor electrónico, mecánico y de comunicaciones que trabajarán de forma integrada, para convertir un simple cinturón de seguridad en un interruptor que se acciona en el momento de abrochar o desabrocharlo.

En la Figura 3.1, se detalla un bosquejo de todos los elementos o dispositivos electrónicos que conforman el prototipo para el control y monitoreo automático del ingreso y salida de pasajeros en un bus interprovincial con alertas SMS, también se verificará la interacción de dichos elementos con el microcontrolador Arduino.

Dentro de todos los dispositivos electrónicos a usarse se adicionará un módulo SIM GPS/GSM, cuyo propósito es enviar un mensaje de texto corto de los eventos que sucedan en ese momento como, por ejemplo: el de avisar si algún pasajero está abordando o saliendo de la unidad, notificar si el usuario está haciendo uso o no del cinturón de seguridad e indicar el estado de la puerta principal. Este mensaje llegara al socio o dueño de la unidad indicando las coordenadas, fecha y hora del lugar donde está ocurriendo dicha eventualidad en ese momento.

Por otro lado, se dispondrá de una base de datos el cual tiene como propósito almacenar todos los datos que lleguen al Arduino, los mismos que podrán ser visualizados a través de una página web en línea, dicha base de datos será implementada en un servidor web, para lograr este propósito se dispondrá de un microcomputador.

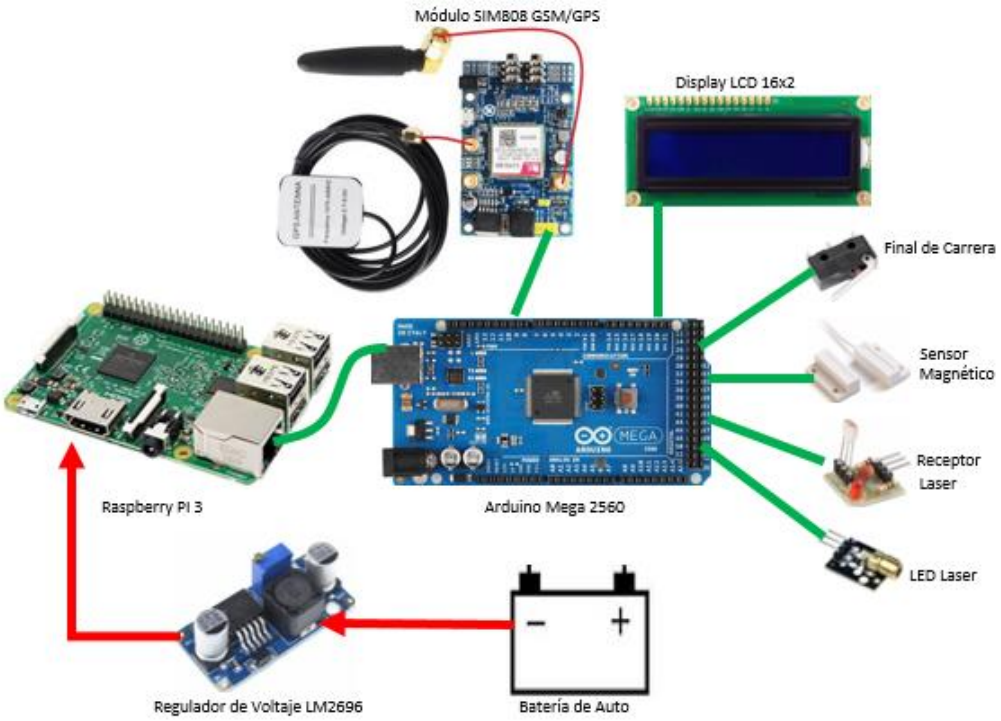


Figura 3.1 Esquema de conexión de los dispositivos electrónicos

Fuente: Elaborado por el autor

3.2 Diagrama de bloques del sistema.

A continuación, se desarrolla un diagrama de bloques, donde se presenta los elementos electrónicos más primordiales a usarse y también la comunicación entre Arduino y Raspberry. En la Figura 3.2 se observa dicho diagrama.

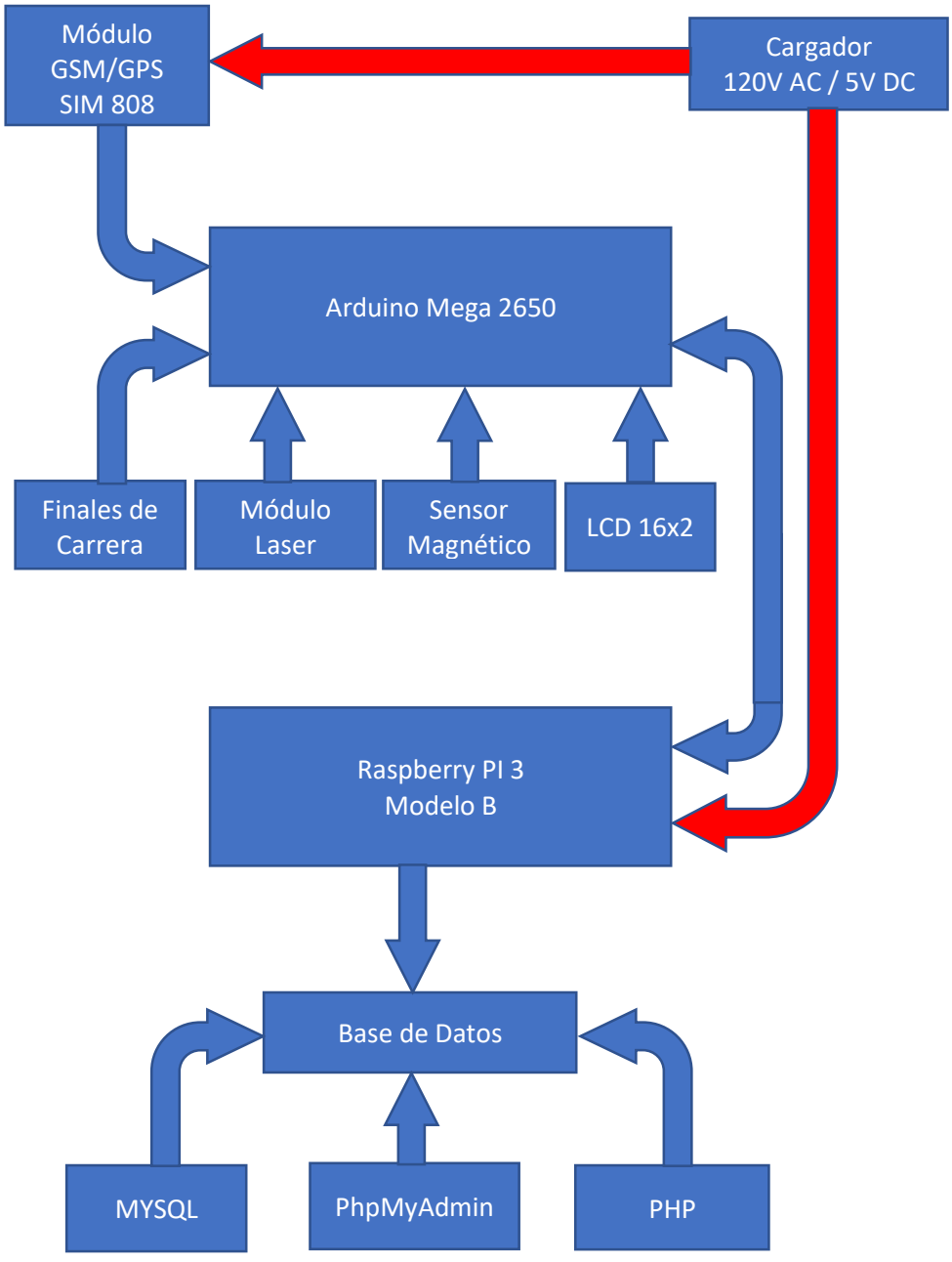


Figura 3.2 Diagrama de bloques del sistema

Fuente: Elaborado por el autor.

3.3 Esquema de flujo.

En el esquema de flujo que se presenta a continuación se explica el modo de funcionamiento del prototipo, partiendo con la inicialización del módulo SIM808 GSM/GPS, cuyo propósito es el de enviar los mensajes cortos al usuario autorizado de las diferentes eventualidades que sucedan dentro de la unidad, el número del destinatario será configurado previamente, este módulo se enlaza directamente al Arduino en los pines 10 y 11. (Transmisión y recepción respectivamente)

Una vez inicializado el módulo SIM808 GSM/GPS o conexión para buscar el posicionamiento actual, el prototipo se encuentra listo para operar y los elementos que se colocará son: un final de carrera por cada asiento, en la puerta del pasillo se instalará un par de módulos laser y finalmente en la puerta principal se implementará un sensor magnético.

Internamente el microcontrolador Arduino Mega 2560, realizará la comparación de si fue accionado o no el sensor magnético, en caso de ser afirmativa la respuesta se ejecuta la orden de enviar un SMS indicando lo sucedió adjunto con las coordenadas, velocidad y al mismo tiempo se almacenará dicho evento en la base datos que se encuentra en el microcomputador.

El módulo laser detecta si la persona o usuario se encuentra ingresando o saliendo de la unidad según la lógica programada en caso de que cumpla cualquiera de las condiciones antes descritas se ejecuta la orden de enviar un SMS indicando lo sucedió adjunto con las coordenadas y al mismo tiempo se almacenará dicho evento en la base datos que se encuentra en el microcomputador.

Mediante la señal emitida por los interruptores finales de carrera se realizará la comparación de si fue accionado o no el cinturón de seguridad, en caso de ser afirmativa la respuesta se ejecuta la orden de enviar un SMS indicando lo sucedió adjunto con las coordenadas y al mismo tiempo se almacenará dicho evento en la base datos que se encuentra en el microcomputador.

Se integrará un *display* LCD 16x2 cuyo propósito será el de mostrar información general del estado actual de los asientos, adicionalmente contará con un panel LED que indicará el número de asiento que ocupará ese momento.

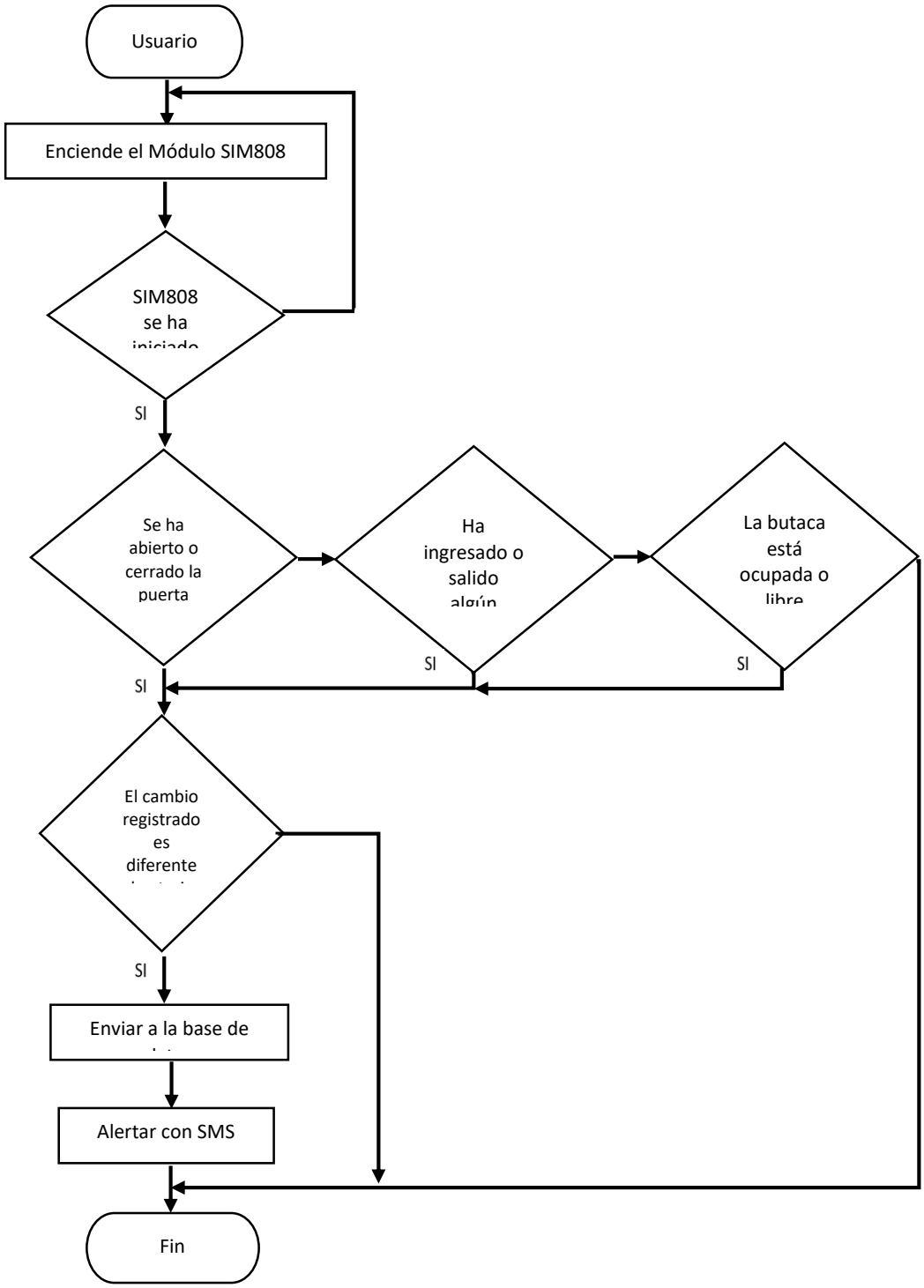


Figura 3.3 Esquema de flujo del sistema

Fuente: Elaborado por el autor

De la misma manera, el beneficiario autorizado puede comprobar las eventualidades sucedidas en el transcurso del recorrido, esto es posible ya que la base de datos se encuentra atada a una página web donde se despliega toda la información, a esta página se podrá acceder con las credenciales autorizadas por el desarrollador, gracias a esta elección se conseguirá controlar el recaudo del dinero que se obtenga al final del día y así evitar posibles fraudes económicos.

3.4 Módulos del prototipo.

El prototipo completo que comprende el control y monitoreo de ingreso y salida de pasajeros con alertas SMS, este conforma de varios en módulos y cada módulo tiene dispositivos electrónicos que hacen posible su funcionamiento. A continuación, se presenta una pequeña lista de los módulos más principales que conforma el proyecto

- Módulo SIM808 GSM/GPS.
- Módulo de control.
- Sensores.
- Comunicación.

3.5 Módulo SIM808 GSM/GPS.

El módulo SIM808 GSM/GPS del prototipo es uno de los elementos principales del prototipo el cual proporciona las coordenadas, velocidad y envía SMS al usuario autorizado para notificar de las eventualidades que ocurran dentro de la unidad durante el recorrido.

3.6 Módulo de control.

Arduino Mega 2560 será el corazón del prototipo, cuyo propósito es de procesar todas las señales que son enviados por los diferentes dispositivos electrónicos conectados al microcontrolador (sensor magnético, modulo laser, final de carrera) este facilita el control y administración de todos los elementos electrónicos conectados hacia el microcontrolador de esta manera garantizar un óptimo funcionamiento del prototipo.

3.6.1 Arduino Mega 2560

Arduino Mega 2560 en un controlador diseñado para proyectos de tipo más robustos, integra un microchip *ATmega*2560, en el cual las funcionalidades del mismo lo hacen muy similar a sus predecesores, la gran diferencia de Arduino Mega con sus predecesoras es el tamaño y la disponibilidad de puertos. Cuenta con una alimentación DC de 7 a 12 voltios, trabaja con un puerto USB Tipo B hembra. En la Figura 3.4 se evidencia la distribución de puertos que dispone este microcontrolador.

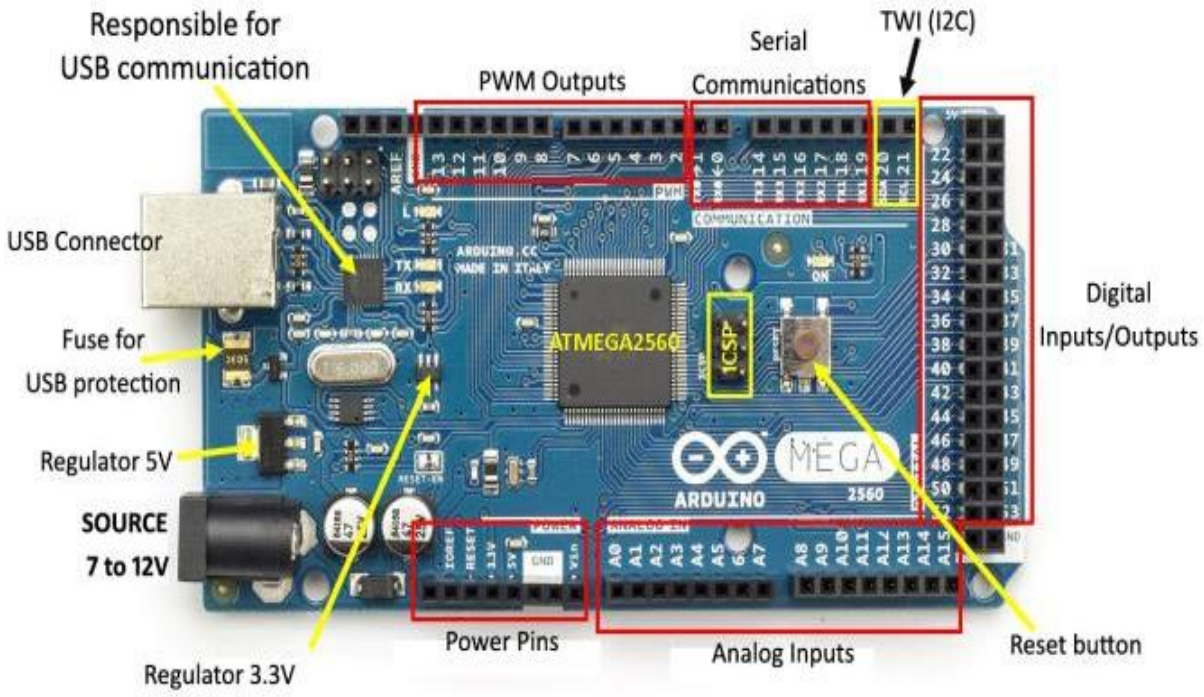


Figura 3.4 Arduino Mega 2560 distribución de pines

Fuente: (Prometec, 2018)

3.6.1.1 Alimentación o fuente de poder

Arduino Mega 2560, se lo puede colocar en funcionamiento a través del puerto USB Tipo B, además suporta un voltaje entre 7-12 V DC a través de una fuente de voltaje continua DC que la proporcionara el Raspberry PI 3, desde uno de los puertos USB que dispone el mismo.

3.6.1.2 Memoria.

Arduino Mega 2560 cuenta con un chip *ATmega* 2560 tiene 256 KB, donde 4 KB corresponden a memoria EEPROM, 8 KB corresponden a la memoria de arranque y 8 KB para memoria RAM esto lo hace superior a sus predecesoras.

3.6.1.3 Puertos de entrada y salida.

Dispone de 54 pines de los cuales 15 de ellos proporcionan una salida PWM y tiene la capacidad de soportar 5 voltios y entregar una corriente de 20mA por cada puerto, también posee 16 puertos analógicos con una resolución de 10 bits y otros puertos como son la de comunicación serial, etc. Arduino Mega 2560 también dispone de puertos para interrupciones los cuales son usados en el desarrollo de diferentes proyectos.

3.6.1.4 Comunicación.

Arduino Mega 2560 cuenta con 4 puertos de comunicación seria y un puerto al cual se puede conectar directamente al computador el puerto *UART TTL* (5V) provee una salida de 5 voltios, el cual cuenta con un puerto transmisión y recepción respectivamente para transmisión entre el módulo SIM808, el programa de Arduino IDE cuenta con una librería de convertir un puerto de propósito general a un puerto de transmisión y recepción de texto plano entre el Arduino y el computador, en este caso solo usaremos para poder visualizar datos a través del puerto serial.

3.6.1.5 Programación.

Arduino posee una interfaz o entorno de programación en su página oficial y es de descarga gratuita, este programa está disponible para Linux, MAC y Windows.

El programa o *software* es de entorno de desarrollo abierto, el cual permite escribir un código que sea compatible un dispositivo que soporte lenguaje de alto nivel, el lenguaje C

es en el cual esta baso Arduino IDE, dentro de este programa también se puede agregar más librerías las cuales ayudan al proceso de programación del prototipo de construcción.

3.6.1.6 Características del microcontrolador.

Dentro de la tabla 3.1 se detallan las características más relevantes del microcontrolador Arduino Mega 2560, en la Figura 3.4 se visualiza la cantidad completa de puertos q posee este microcontrolador.

Tabla 3.1 Principales características Arduino Mega 2560

Características Arduino Mega 2560	
Microcontrolador:	*Atmega* 2560
Alimentación de entrada:	7V
Puertos digitales:	54
Puertos PWM	15
Pines analógicos:	16
Valor de corriente proporcionada en cada pin:	20 Ma
Valor de corriente proporcionada en el pin 3.3V:	50 Ma
Velocidad de reloj:	16 MHz
Memoria EEPROM:	4 Kb
Memoria SRAM:	8 Kb
Memoria flash:	256 Kb

Fuente: (Prometec, 2018)

3.7 Sensores.

Son elementos electrónicos que tienen la capacidad de detectar una capacidad física o química del entrono o medio y puede transformarla en una señal eléctrica para su respectivo procesamiento.

3.7.1 Módulo sensor laser KY-008.

El módulo sensor laser KY-008 está constituido de un transmisor (TX) y receptor (RX) el cual transmite luz láser de 650 nm de longitud de onda al receptor, al momento de obstruir esta luz, se emite una señal cuando a través del mismo indicando o ejecutando una acción programada.

Figura 3.5 Módulo sensor laser KY-008

Fuente: (Vistronica, 2018)

Características principales:

- Voltaje de funcionamiento: 5V
- Tamaño del PCB: 15 * 24mm
- Longitud de onda: 650 nm (Rojo)
- Profundidad: 8 mm
- Altura: 24 mm
- Ancho: 15 mm
- Peso: 2.2 g

3.7.2 Sensor magnético de puerta MC-38.

El sensor magnético de puerta MC-38 permite detectar si una puerta está abierta o cerrada, el sensor y el imán cierran o abren el circuito de acuerdo si están cerca uno del otro, estos sensores son muy utilizados en domótica o cundo se necesita construir un sistema de seguridad complejo, es de fácil integración con Arduino y Raspberry PI, este sensor será ubicado en la puerta o acceso principal del autobús.

Figura 3.6 Sensor magnético de puerta MC-38

Fuente: (Tecmikro, 2018)

Características principales:

- Producto: MC38
- Corriente máx.: 0.5A
- Voltaje máx.: 100V
- Distancia de activación: 15-25mm
- Longitud de cable: 25cm
- Material: Plástico Blanco ABS
- Dimensiones: 34 × 41 × 6.5 mm

3.7.3 Sensor magnético de puerta MC-38.

El sensor magnético de puerta MC-38 permite detectar si una puerta está abierta o cerrada, el sensor y el imán cierran o abren el circuito de acuerdo si están cerca uno del otro, estos sensores son muy utilizados en domótica o cundo se necesita construir un sistema de seguridad complejo, es de fácil integración con Arduino y Raspberry PI, este sensor será ubicado en la puerta o acceso principal del autobús.

Figura 3.7 Interruptor final de carrera

Fuente: (Vistronica, 2018)

Características principales:

- Incluye micro pulsador
- 1A - 125V – AC/DC
- 3 pines
- Color negro, blanco y plateado
- Tamaño: 1,2 x 0.5 cm

3.8 Comunicación.

Para establecer la conexión entre Arduino y Raspberry PI comúnmente se utiliza un cable USB, estos dispositivos usan el puerto serial para poder comunicarse entre sí, permitiendo así él envió y recepción de datos la conexión se puede apreciar en la Figura 3.8.

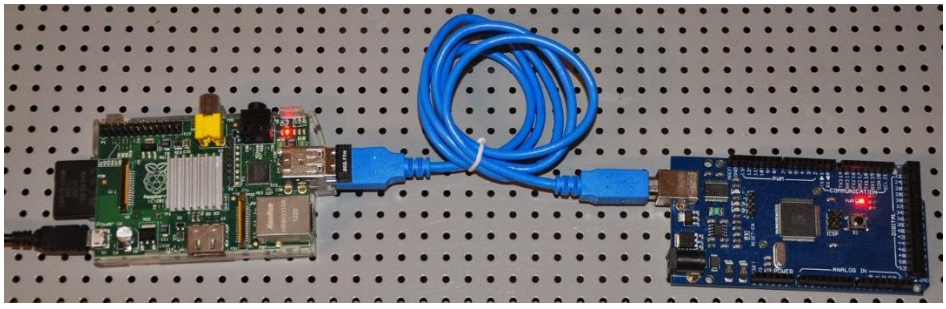


Figura 3.8 Interconexión entre Raspberry PI y Arduino

Fuente: Elaborado por el autor

3.9 Programa o *Software*.

Para escribir codificación del programa en el microcontrolador Arduino Mega 2560, realizar el diseño de piezas que confirmaran la maqueta del prototipo y establecer la conexión de la página web es recomendable utilizar de programas específicos de cada uno de los elementos electrónicos mencionados anteriormente.

3.9.1 Arduino IDE.

Es un programa libre y de código abierto el cual permite escribir cualquier programación que soporte lenguaje de alto nivel es decir que se encuentre basado en el

lenguaje C, de la misma manera con el entorno de Arduino IDE se puede crear librerías y usar librerías que en muchas ocasiones son recomendadas por el fabricante.

La plataforma de Arduino IDE ofrece la capacidad de desarrollar o construir un número infinito programas e incluso mirarlos a través del puerto serial que dispone.

3.9.2 Proteus.

EL programa es una de las herramientas más utilizadas para la simulación de circuitos electrónicos, contiene una gran cantidad de librerías, elementos electrónicos, herramientas para realizar pruebas como multímetro virtual, osciloscopio, etc., además la de poder imprimir un diagrama circuital de todos los elementos electrónicos que contemplan este prototipo, cuya finalidad será la de dar a conocer la forma de cómo realizar las diferentes conexiones.

La herramienta informática Proteus se usará para construir el diagrama electrónico del proyecto, con todos los dispositivos electrónico que forman parte del prototipo.

3.9.3 Ares.

Ares es un programa para simular de enrutado, ubicación y edición de los dispositivos electrónicos, su propósito es el de crear o fabricar placas de circuito impreso permitiendo editar generalmente las capas superficialmente y de soldadura.

El programa Ares será utilizado para la construcción e impresión de cierta cantidad de pistas electrónicas indispensable dentro de las conexiones a realizarse en el prototipo, en función de sus dispositivos electrónicos constitutivos como son los finales de carrera, sensor magnético de puerta, sensor laser, panel de LED´s y LCD.

3.9.4 Sublime Text.

Sublime Text es un editor de texto y editor de código fuente, ligero, multiplataforma y cuenta con un abundante catálogo de *plugins*.

El programa Sublime Text será utilizado para la elaboración del código fuente del diseño de las páginas web e interconexión entre la base de datos y la página web.

3.10 Aspectos técnicos del producto.

La construcción del prototipo dispondrá de un enlace, entre Arduino y Raspberry PI a través de un cable USB, a través del mismo se transmitirá los diferentes datos que envíen los sensores hacia la base de datos implementada en el microcontrolador.

Se emplea un final de carrera por cada cinturón de seguridad ubicado en cada asiento del autobús, mediante el cual se accionará cuando el usuario se coloque o quite el cinturón de seguridad, enviando así una señal al Arduino el cual la procesará y enviará tanto a la base de datos como al usuario final vía SMS, indicando el número de la butaca que se ocupó, con las coordenadas, fecha y hora de la eventualidad.

El prototipo dispone de un módulo sensor laser, este trabajará como un interruptor en el cual se accionará cada vez que se obstruya la luz láser, para poder detectar si el usuario sube o baja de la unidad se adicionará un módulo laser en cual, se podrá detectar el sentido en el que está movilizándose el usuario dentro de la unidad y cuando cumpla la condición de ingresando pasajero este enviara una señal al Arduino y posteriormente un SMS al usuario final indicando las coordenadas, fecha y hora del evento suscitado.

Mediante un teléfono móvil o un ordenador se podrá acceder a la base de datos, contendrá todos los registros y eventos acontecidos durante el recorrido que ha realizado la unidad de transporte, con el propósito de que esta información sea analizada por los socios o usuarios correspondientes de la compañía.

3.11 Análisis de costos para el desarrollo del proyecto.

El proyecto se desarrolló a través de los objetivos específicos preestablecidos, se deberá establecer un estudio de costos el mismo que servirá de base para indagar diferentes dispositivos electrónicos que se acoplen de forma competente a las necesidades del proyecto, los dispositivos o elementos electrónicos deberán cumplir con los requerimientos, funciones y disponibilidad inmediata en el mercado sea asequible para su desarrollo.

3.11.1 Placa de control.

La placa que se usará será un microcontrolador Arduino Mega 2560, y con la ayuda del programa Arduino IDE se procederá a realizar la programación de las diferentes acciones a realizarse en el prototipo.

Para la elección de microcontrolador adecuado se realizó un previo análisis de las opciones que se encontraban en el mercado electrónico donde se comparó Arduino Mega 2560 y Arduino UNO donde sus características son casi similares y su gran diferencia radica en el número puertos, memoria, etc.

La primera opción que se seleccionó fue el microcontrolador Arduino UNO, ya que este microcontrolador cuenta con las características necesarias para construir o desarrollar proyectos pequeños en donde la limitación de puertos no sea necesaria, puertos TX-RX, posee puerto PWM, puertos analógicos y puertos digitales, también cuenta con la opción de alimentar al microcontrolador de manera directa, al poseer limitación de puertos no se usará la misma ya que para la realización de este proyecto se necesita más puertos de propósito general para conectar los diferentes elementos electrónicos a la placa.

Como alterativa, es seleccionar el microcontrolador Arduino Mega 2560, en el cual este cuenta con mejores características y ventajas que el Arduino UNO, mayor memoria, mayor cantidad de puertos y cumple con los puertos necesarios para el desarrollo del proyector.

Después de indagar acerca de los microcontroladores es necesario seleccionar el microcontrolador Arduino Mega 2560, este dispone de una mayor cantidad puertos digitales, puertos analógicos, etc. de la misma manera su memoria se duplica en comparación a la del Arduino UNO y su precio es sumamente asequible.

En la tabla 3.2, se detalla una pequeña comparación de precios de los microcontroladores analizados con anterioridad.

Tabla 3.2 Precios microcontrolador Arduino

Precios de microcontroladores			
	Arduino Mega 2560	**Arduino UNO**	
Precio	$17	$10	

Fuente: (Tecmikro, 2018)

En conclusión, se elige el microcontrolador Arduino Mega 2560, ya que cuenta con los requerimientos necesarios para el desarrollo de prototipo.

3.11.2 Microcomputador Raspberry PI.

Existen diferentes versiones de Raspberry PI que son aptos para el proyecto, estos cumplen con especificaciones y características similares, además de que sus precios son casi similares uno del otro.

En la tabla 3.3, se detalla la comparación en precios entre Raspberry PI 2 y Raspberry PI 3.

Tabla 3.3 Precios de Raspberry PI

Comparación de precios			
	Raspberry PI 2	**Raspberry PI 3**	
Precio	$63	$65	

Fuente: (Raspberry PI, 2018)

Se seleccionó el microcomputador Raspberry PI 3, el cual cuenta con una tarjeta de red inalámbrica integrada eliminando así la necesidad de conectarse de forma cableada a la red, además que dispone de los requisitos mínimos para implementar un pequeño servidor de base de datos.

3.11.3 Módulo GSM/GPS.

El módulo SIM808 ofrece, además de las funcionalidades de envío y recepción de datos GSM/GPRS (la de los teléfonos móviles 2g), la tecnología GPS de navegación por satélite. A comparación del módulo SIM900 el cual posee adicionalmente 12 pines GPIO de propósito general para aplicaciones maquina a máquina. Cualquiera de estos módulos funciona utilizando una tarjeta SIM, en el cual será capaz de enviar y recibir llamadas y SMS, conectarnos a Internet y conocer nuestras coordenadas y el horario UTC (Tiempo Universal Coordinado) con las características y necesidades deseadas, además de sus precios similares.

En la tabla 3.4, se detalla la comparación de precios entre los diferentes módulos (Módulo SIM808 y Módulo SIM900).

Tabla 3.4 Precios de los módulos SIM GSM/GPS

COMPARACIÓN DE PRECIOS			
	Módulo SIM808	**Módulo SIM900**	
Valor	$34	$44	

Fuente: (Prometec, 2018)

Se elegirá el módulo SIM808 el cual cuenta y cumple con los requerimientos necesarios para continuar con el desarrollo del prototipo, con un precio menor, incluye una antena GPS y consume una cantidad de corriente baja de tan solo de 100 mA.

3.11.4 Final de Carrera.

En el mercado existe una gran variedad de finales de carrera, de todo tamaño y precio, para la realizar este proyecto y que se acople de mejor manera a los cinturones de seguridad

se optara por los finales de carrera más pequeños que existe en el mercado donde su precio es sumamente bajo y sus dimensiones físicas son pequeñas.

En la tabla 3.5, se detalla la diferencia de precios de dos finales de carrera pequeños en comparación a los de mayor magnitud física.

Tabla 3.5 Precios de final de carrera

Comparación de precios			
	Final de carrera pequeño	**Final de carrera grande**	
Precios	$0,15	$0,25	

Fuente: (Vistronica, 2018)

3.11.5 Módulo Laser.

A diferencia del diodo laser emisor de luz y una fotorresistencia el módulo laser KY-008, trae en si una pequeñísima placa en la cual sus componentes ya vienen integrados, y son fáciles de conectar, su precio es sumamente bajo a diferencia de adquirir un diodo laser y una fotorresistencia.

En la tabla 3.6, se detalla la diferencia de precios de un módulo KY-008 y un diodo laser más una fotorresistencia.

Tabla 3.6 Precios de módulo laser y diodo laser más fotorresistencia

Comparación de precios			
	Módulo Laser KY-008	**Diodo laser y fotorresistencia**	
Valor	$1,00	$1,70	

Fuente: (Dualtronica, 2018)

Se elegirá el modulo laser KY-008 porque sus componentes ya vienen integrados en su propia placa y no es necesario colocar resistencias o capacitores para optimizar su funcionamiento y otra gran ventaja es que su precio es sumamente bajo.

3.11.6 Presupuesto del proyecto.

Después de haber determinado la cantidad de dispositivos a usarse en todo el proyecto, se deberá realizar un cálculo del costo total del prototipo, para el control y monitoreo automático del ingreso y salida de pasajeros en un bus interprovincial con alertas SMS, dentro de la tabla 3.7 se presenta el cálculo total del prototipo.

Todos los elementos electrónicos involucrados durante el desarrollo del prototipo fueron considerados y analizados para la construcción del prototipo.

Para seleccionar los dispositivos electrónicos se tomó en consideración varios aspectos como son el tamaño, precio, complejidad, etc.

También se tomó en consideración que exista gran disponibilidad de dispositivos electrónicos dentro del mercado electrónico, ya que por varias razones se tenga que reemplazar el dispositivo y así puedan ser encontrados con gran facilidad dentro del mercado nacional.

A continuación, se presenta en detalle el costo total del proyecto, con los valores de cada uno de los dispositivos a usarse.

Tabla 3.7 Costo total del proyecto

Cantidad	Elemento o Material	Valor Unitario ($)	Valor total ($)
1	Módulo Arduino Mega 2560	17	17
1	Módulo Raspberry PI 3	63	63
1	Módulo GPS/GSM SIM808	34	34
8	Final de carrera	0,15	1,20
1	Interruptor magnético para puertas o ventanas	0,60	0,60
2	Módulo Laser KY-008	1	2
1	Diseño y elaboración de la página web del prototipo	40	40
1	Diseño y elaboración de la base de datos	20	20

Cantidad	Elemento o Material	Valor Unitario ($)	Valor total ($)
1	Diseño y elaboración maqueta de un autobús con 8 pasajeros de capacidad	40	40
1	Paquete de cables de conexión Arduino, de 10 cm Macho-Hembra	3	3
	Total		220,8

Fuente: Elaborado por el autor

3.12 Análisis de tiempos.

Para realizar el análisis de tiempos se debe realizar una descripción minuciosa del cronograma que se presente durante las etapas que abarcan desarrollo total del presente proyecto, para la primera fase se definirá la cantidad de elementos electrónicos que formaran parte del prototipo de control y monitoreo automático del ingreso y salida de pasajeros en un bus interprovincial con alertas SMS, alguno de estos elementos electrónicos que formaran parte del desarrollo del prototipo son: de tipo mecánico, y de *software* libre a usarse, etc., Los dispositivos electrónicos deben cumplir con ciertos requerimientos mínimos, los elementos del prototipo se seleccionan de acuerdo con el costo y beneficio que disponga. La duración de esto tomó 16 días, desde el sábado 28 de abril hasta el lunes 14 de mayo del 2018.

Una vez que se hayan elegido los dispositivos electrónicos a usarse se deberá realizar un diagrama de conexiones de todos los elementos para su respectivo funcionamiento, determinar el lugar adecuado de los finales de carrera, interruptor magnético, sensor laser y establecer el lugar adecuado del prototipo dentro de la unidad de transporte, la duración de esta tarea es de 8 días desde el martes 15 de mayo hasta el martes 22 de mayo.

En la siguiente fase comprende la construcción del programa en Arduino en donde se escribirá el código de programa que permita actuar según la lógica establecida por los elementos electrónicos que componen el prototipo, establecer el enlace entre el modulo GPS/GSM y Arduino, depurar errores que se vayan presentando y unificar los elementos electrónicos para verificar su correcto funcionamiento esto se logrará en un tiempo de 19 días desde el miércoles 23 de mayo hasta el domingo 10 de junio.

En la siguiente fase comprende a establecer el enlace entre Raspberry PI y Arduino en donde en lo primero a realizar será instalar el sistema operativo Raspberry PI en el microcomputador, se deberá de realizar pruebas de comunicación en donde se envíen datos que provengan de los diferentes sensores que lleguen al Arduino y los reciba en un script escrito en Python, etc., Para completar toda la tarea el tiempo estimado será de 8 días desde el lunes 11 de junio hasta el lunes 18 de junio del 2018.

Una vez establecido la programación de todos los dispositivos electrónicos y su respectiva comunicación con Raspberry PI se deberá proceder a instalar y validar el correcto funcionamiento de un servidor APACHE, PHP, PhpMyAdmin y MyQSL, después se deberá crear las respectivas tablas que conformaran la base de datos del prototipo cuya función principal es recolectar todos los datos de los eventos que vayan aconteciendo en el transcurso del viaje del autobús, en realizar todas estas tareas tomará 9 días desde el martes 19 de junio hasta el miércoles 27 de junio del 2018.

Para realizar la construcción de la maqueta y demostrar el funcionamiento respectivo del prototipo, se deberá instalar todos los elementos electrónicos dentro de ella adaptando de mejor manera a las dimensiones de la maqueta tomará 14 días desde el jueves 28 de junio hasta el miércoles 11 de julio del 2018.

Ya en última fase tarea se deberá realizar pruebas de funcionamiento de todo el prototipo con todos los dispositivos que involucran el mismo, así como la de poder depurar los posibles errores que se presenten en las pruebas a realizarse, esta tarea tendrá una duración de 5 días desde el jueves 12 de julio hasta el lunes 16 de julio del 2018. En el anexo 7 se presenta el cronograma completo con todas las actividades a realizarse para la elaboración del proyecto.

3.13 Ventajas del producto.

El uso de un sistema de control y monitoreo de pasajeros hace que la recaudación de dinero al final del día sea más real y exista menos adulteraciones por terceros y falsos reportes sean entregados al dueño o socio de la unidad.

Se podrá obtener alertas de las eventualidades programadas dentro de la unidad, así como la de conocer su ubicación, fecha y hora en la sube o baja el usuario de bus interprovincial, dichas alertas pueden se programadas para que las reciba el socio o dueño de la compañía o según sea necesario.

El dispositivo cuenta con una base de datos integrada en la cual se almacenará todas las eventualidades ocurridas en la unidad de transporte, con la cual a futuro podría ser analizada para mejoras de la compañía.

Además, servirá para poder controlar que los usuarios este haciendo uso del cinturón de seguridad, para de esta manera poder cumplir con la ley orgánica de tránsito y transporte terrestre, lo cual podría ocasionar multas que sería perjudicial para la compañía.

CAPÍTULO 4

IMPLEMENTACIÓN

4.1 Desarrollo

Para el desarrollo del prototipo de control y monitoreo automático del ingreso y salida de pasajeros en un bus interprovincial con alertas SMS, fue indispensable la compra de los materiales electrónicos necesarios para su elaboración. Estos materiales son de fácil adquisición dentro del mercado electrónico, para empezar con la construcción del proyecto es necesario contar con una maqueta, la cual servirá como base para iniciar, después se procederá con la implementación de la parte de control y finalmente con el desarrollo de la programación del mismo.

Maqueta del prototipo. – Se simulará un modelo de bus interprovincial con la capacidad de 8 pasajeros. En cada butaca o asiento se incluirá un cinturón de seguridad de forma obligatoria, el panel de control se encontrará en la cabina del conductor.

Parte de control. - Estará compuesta por la electrónica que abarca el microcontrolador Arduino Mega 2560, la cual es la parte central del prototipo, además de controlar todos los dispositivos o componentes que conforman el mismo, los dispositivos electrónicos como son los sensores finales de carrera cuya función principal es la de verificar cuando un cinturón de seguridad se haya accionado, también se dispondrá de un sensor laser infrarrojo en la puerta hacia el pasillo, este registra o detecta el movimiento el ingreso o salida de pasajeros, además se incluirá un panel de LED que permitirá observar que butaca o asiento está siendo ocupado.

Parte de *Software*. - Comprende la visualización de información en tiempo real desde un dispositivo con acceso a internet, la página se encontrará a alojada en un

microcomputador y también servirá para almacenar todos los eventos que sucedan durante el viaje o recorrido de la unidad de transporte, este mini servidor se construirá con varias herramientas totalmente gratis, que son asequibles desde la web.

4.1.1 Diagrama de bloques.

En la Figura 4.1 se presenta un esquema o diagrama de bloques del de funcionamiento del prototipo para el control y monitoreo automático del ingreso y salida de pasajeros en un bus interprovincial con alertas SMS.

Cuenta con un sensor magnético ubicado en la puerta principal de ingreso, cuya función es detectar el momento en que la puerta ha sido abierta para que los usuarios puedan ingresar a la unidad. Para que los usuarios puedan llegar a la butaca o asiento de preferencia los usuarios deberán pasar de forma obligatoria por la puerta secundaria, en dicha puerta secundaria existirá un sensor laser que al momento de cortar la luz este detectara que está subiendo o bajando el usuario de la unidad de transporte, en las butacas o asientos existirá un cinturón de seguridad el cual será alterado para insertar interruptor electrónico, cuya función será la de enviar una señal en el momento que este sea accionado por el usuario.

Las señales que envíen los diferentes sensores instalados en la unidad serán procesadas por el microcontrolador Arduino Mega, este a su vez los enviará a una base de datos previamente instalada en un microcomputador Raspberry en el cual serán almacenados para propósitos futuros de la compañía o socio.

Se tendrá la posibilidad de visualizar la información en tiempo real con tan solo acceder a sitio web, en el cual se obtendrá información detallada de los diferentes eventos que ocurran en la unidad, además, cuenta con un sistema de SMS cuyo propósito es notificar al socio de la unidad de ciertos eventos que el usuario requiera.

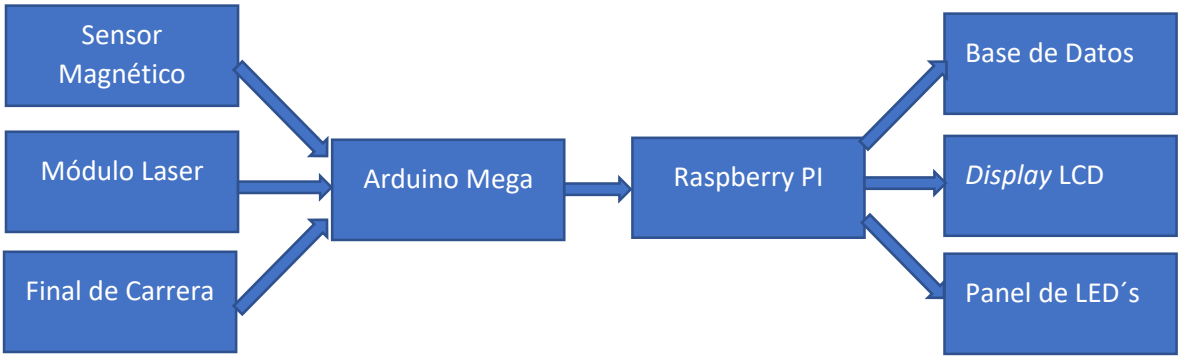


Figura 4.1 Diagrama de bloques del esquema de funcionamiento del prototipo

Fuente: Elaborado por el autor

Dentro de la Figura 4.2 se puede observar el esquema o diagrama de control para el funcionamiento del prototipo del presente proyecto para el control y monitoreo automático del ingreso y salida de pasajeros en un bus interprovincial con alertas SMS, en esta figura se visualiza los elementos constituyen dicho prototipo.

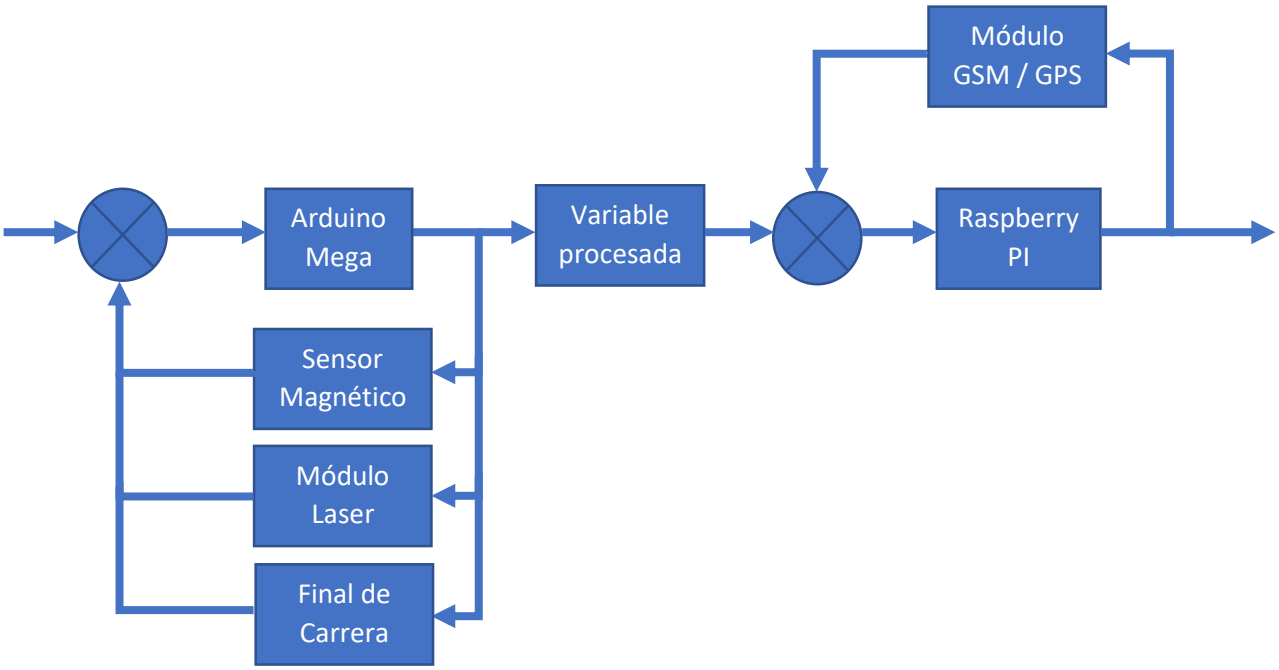


Figura 4.2 Diagrama de control

Fuente: Elaborado por el autor

4.1.2 Desarrollo del programa para lectura de datos en Arduino.

La programación se realizó con la herramienta informática Arduino IDE que es un programa libre, en el será posible escribir el código de programación de manera fácil y también se deberá instalar ciertas librerías recomendadas por el fabricante, para el correcto funcionamiento de cada uno de los dispositivos electrónicos que se conecten hacia el microcontrolador, como son los interruptores finales de carrera, el módulo SIM808 GSM/GPS, el módulo laser y el sensor magnético, también se tienes dispositivos o interfaces de salida como son la pantalla LCD y el panel de pasajeros que está compuesto por LED's.

4.1.3 Entorno de Arduino IDE.

La instalación de Arduino IDE es sumamente sencilla y en esta ocasión no se la detallará como tal ya que incluso en la página principal de Arduino viene a detalle los pasos a seguir para poder instalarlo en la computadora.

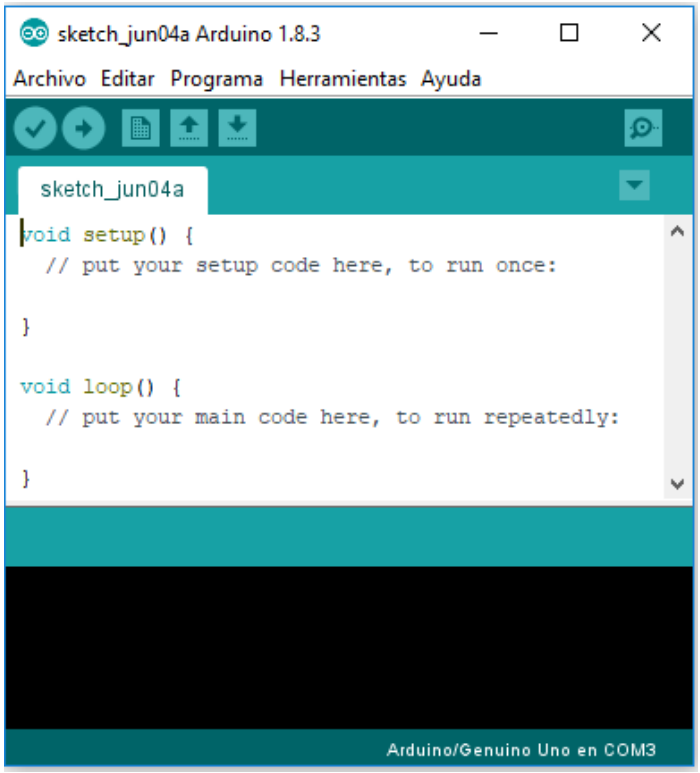


Figura 4.3 Interfaz de Arduino IDE

Fuente: Elaborado por el autor

4.1.4 Código para recepción de coordenadas GPS y envió de SMS.

El desarrollo del código de programación para la recepción de coordenadas GPS y envió de SMS se lo realizó de la siguiente manera, en el cual se incluyen las librerías de DFRobot_sim808.h y SoftwareSerial.h recomendadas por el fabricante del módulo SIM808 GSM/GPS.

```
#include <DFRobot_sim808.h>
#include <SoftwareSerial.h>

#define PIN_TX    10
#define PIN_RX    11

SoftwareSerial mySerial(PIN_TX,PIN_RX);
DFRobot_SIM808 sim808(&mySerial);

char MESSAGE[300];
char lat[12];
char lon[12];

#define PHONE_NUMBER "0990257279"
#define MENSAJE_PUERTA "LA PUERTA SE ABRIO"
#define MENSAJE_PUERTC "LA PUERTA SE CERRO"
#define MENSAJE_PASAJERO "INGRESO PASAJERO"
#define MENSAJE_PASAJERO_S "SALIENDO PASAJERO"

#define MESSAGE_LENGTH 160
char message[MESSAGE_LENGTH];
int messageIndex = 0;
char phone[16];
char datetime[24];
```

Figura 4.4 Líneas de código para SIM808 GSM/GPS

Fuente: Elaborado por el autor

En la anterior imagen se puede apreciar que se están incluyendo las librerías antes mencionadas, también se están definiendo los pines de transmisión y recepción para enviar y recibir mensajes de texto corto, se define una variable de tipo *char* para enviar el mensaje generado por cualquier evento y también dos variables adicionales para enviar las coordenadas en latitud y longitud.

Se define un nombre a ciertos valores constantes, las constates definidas en Arduino no utilizan memoria del programa en el microprocesador por lo que es de gran ayuda al momento de desarrollar el código de programación.

Una vez que se haya configurado los parámetros iniciales se define la primera función a ejecutarse en el microcontrolador Arduino.

```
void setup()
{
  while(!sim808.init()) {
    delay(1000);
    lcd.clear();
    lcd.setCursor(0,0);
    lcd.print("SIM808:");
    lcd.setCursor(0,1);
    lcd.print("ERROR CONEXION");
    Serial.println("ERROR CONEXION");
  }

  if( sim808.attachGPS()){
    lcd.clear();
    lcd.setCursor(0,0);
    lcd.print("GSM:INICIADO");
    lcd.setCursor(0,1);
    lcd.print("GPS:INICIADO");
    delay(2000);
    }
  else {
    lcd.clear();
    lcd.setCursor(0,0);
    lcd.print("GPS:ERROR");
}
```

Figura 4.5 Líneas de código para verificar conexión de módulo SMI808

Fuente: Elaborado por el autor

Se verifica que mientras no esté encendido el módulo SIM 808 GSM/GPS, envié un mensaje de “Error de Conexión” y se lo visualice en el *display* LCD, mientras que cuando se esté inicializando el módulo SIM 808 GSM/GPS envié un mensaje de “GSM: Iniciado” y “GPS: Iniciado” y en caso contrario se visualice “GPS: Error”, dicho código de programación permite el uso del prototipo solo cuando el módulo SIM808 GSM/GPS este operativo.

4.1.5 Código para lectura del sensor magnético en la puerta principal.

Ya escrito el código de programación para el uso del módulo GSM/GPS, se deberá proceder a desarrollar el programa para los diferentes sensores que se conecten a Arduino Mega, como son finales de carrera, módulo laser y sensor magnético.

El desarrollo del código de programación para la lectura del sensor magnético se lo realizó de manera en que cuando se abra o cierre la puerta envié una señal al microcontrolador y este a su vez realice una acción determinada o programada.

```
byte sensor_mag=46;
byte val_sensor_mag = 0;
byte estadoBotonAnt = 0;

void setup() {
pinMode(sensor_mag,INPUT_PULLUP);
}

void loop() {
   val_sensor_mag=digitalRead(sensor_mag);
    if(val_sensor_mag==HIGH && estadoBotonAnt == LOW){
      enviar_mensaje1();
    }
    if(val_sensor_mag==LOW && estadoBotonAnt == HIGH){
      enviar_mensaje3();
    }
    estadoBotonAnt = val_sensor_mag;
}
```

Figura 4.6 Líneas de código para sensor magnético

Fuente: Elaborado por el autor

Se define una variable de tipo entero para el sensor magnético en el puerto 46 del microcontrolador, se configura dicho puerto como entrada y se hace uso de las resistencias pull-up integradas en el chip de Arduino Mega, esto es de gran utilidad y más efectivo que el uso de componentes externos ahorrando así la colocación de una resistencia y un capacitor al momento de desarrollar la placa electrónica

El sensor magnético se inicializa en cero y mientras ocurra un cambio al anterior valor este cambiará por una sola vez, es decir, si la puerta se abre o se cierra se enviará un mensaje

a la base de datos y un SMS al usuario indicando lo sucedido por una sola vez. A continuación, se presenta el código de programación para enviar dicho mensaje.

En las líneas que se describen a continuación también se envía a través del puerto serial un mensaje que será almacenado en la base de datos, en el que se incluirán las coordenadas de donde ocurrió el evento, e incluso detectará si el autobús está en movimiento o se detuvo.

```
void enviar_mensaje1(){
      Serial.print("PUERTA ABIERTA     ");

      float la = sim808.GPSdata.lat;
      float lo = sim808.GPSdata.lon;

      dtostrf(la, 8, 6, lat);
      dtostrf(lo, 8, 6, lon);

      Serial.print(sim808.GPSdata.lat, 7);
      Serial.print(" ");
      Serial.print(sim808.GPSdata.lon, 7);
      Serial.print(" ");
      if (sim808.GPSdata.speed_kph < 0.3){
        Serial.println("DETENIDO");
        }
      else {
        Serial.println("MOVIMIENTO");
        }
      delay(1000);
      sim808.sendSMS(PHONE_NUMBER,MENSAJE_PUERTA);
      delay(1000);
      //Serial.println("MENSAJE ENVIADO LA PUERTA SE ABRIO");
}
```

Figura 4.7 Líneas de código para enviar alerta de puerta

Fuente: Elaborado por el autor

4.1.6 Código para lectura del sensor laser en la puerta del pasillo.

Para detectar que un pasajero suba o baje de la unidad se usara un módulo laser (diodo laser y receptor LDR) en el cual el receptor tiene un valor cuando la luz incide sobre ella, si por algún evento esta luz se ve interrumpida el valor de resistencia cambia, esta variación es detectada por el microcontrolador, se configura dichos receptores en los puertos 47 y 48, a su vez se declarara como valores enteros como se muestra en la siguiente figura.

```
int sensor_e1=47;
int sensor_e2=48;
int val_sensor_e1;
int val_sensor_e2;

void loop() {
    val_sensor_e1=digitalRead(sensor_e1);
    val_sensor_e2=digitalRead(sensor_e2);
    if(val_sensor_e1==LOW){
      if(val_sensor_e2==LOW){
        enviar_mensaje2();}
}
}
```

Figura 4.8 Líneas de código para módulo laser

Fuente: Elaborado por el autor

Se detalla a continuación el mensaje que enviará tanto a la base de datos como al usuario a través de un SMS indicando si el pasajero está subiendo o bajando de la unidad e transporte con sus respectivas coordenadas.

```
void enviar_mensaje2(){
      Serial.print("INGRESO PASAJERO   ");

      float la = sim808.GPSdata.lat;
      float lo = sim808.GPSdata.lon;

      dtostrf(la, 8, 6, lat);
      dtostrf(lo, 8, 6, lon);

      Serial.print(sim808.GPSdata.lat, 7);
      Serial.print(" ");
      Serial.println(sim808.GPSdata.lon, 7);

      delay(1000);
      sim808.sendSMS(PHONE_NUMBER,MENSAJE_PASAJERO);
      delay(1000);
}
```

Figura 4.9 Línea de código para enviar alerta SMS de ingreso de pasajero

Fuente: Elaborado por el autor

4.1.7 Código para lectura los interruptores de final de carrera.

Para detectar que un pasajero este ocupando una butaca o asiento se tendrá instalado en el cinturón de seguridad un interruptor (fin de carrera), este a su vez enviara un mensaje

a la base de datos a través de la comunicación serial y un SMS al usuario, indicando las coordenadas de donde sucedió dicho evento, estos finales de carrera se configurarán desde los puertos 22 al 29, como entrada y a su vez se hace uso de las resistencias pull-up integradas en el chip de Arduino Mega.

En la siguiente figura se evidencia dicha configuración, donde se declara los puertos a usarse como variables enteras, también se puede verificar la configuración de los puertos como entrada y la ejecución de un *loop* para la lectura de los interruptores.

```
void setup()
{
  pinMode(FCA1,INPUT_PULLUP);
  pinMode(FCA2,INPUT_PULLUP);
  pinMode(FCA3,INPUT_PULLUP);
  pinMode(FCA4,INPUT_PULLUP);
  pinMode(FCA5,INPUT_PULLUP);
  pinMode(FCA6,INPUT_PULLUP);
  pinMode(FCA7,INPUT_PULLUP);
  pinMode(FCA8,INPUT_PULLUP);
}

void loop()
{
  if (sim808.getGPS()) {

    leer_FC();
  }
}
```

Figura 4.10 Líneas de código para declarar finales de carrera como entradas

Fuente: Elaborado por el autor

La lectura de los interruptores se lo realiza en un *loop* diferente, donde con la ayuda de un contador y variables locales se podrá leer el estado de los interruptores finales de carrera y determinar cuántos asientos están siendo ocupados o libres.

```
void leer_FC(){
  val_FCA1=digitalRead(FCA1);
  val_FCA2=digitalRead(FCA2);
  val_FCA3=digitalRead(FCA3);
  val_FCA4=digitalRead(FCA4);
  val_FCA5=digitalRead(FCA5);
  val_FCA6=digitalRead(FCA6);
  val_FCA7=digitalRead(FCA7);
  val_FCA8=digitalRead(FCA8);

int v[] = {val_FCA1, val_FCA2, val_FCA3, val_FCA4,
val_FCA5,val_FCA6, val_FCA7, val_FCA8};
int n=0;
int i=0;
int libres;

for(i=0;i<=7;i++){
  if(v[i]==0){
    n++;
  }
}
libres=8-n;
```

Figura 4.11 Líneas de código para leer el estado de los finales de carrera

Fuente: Elaborado por el autor

4.1.8 Código de programación para mostrar la información en *display* LDC y panel de pasajeros.

Dentro del *loop* que se designó para leer el estado de los finales de carrera se incluirá también las líneas de código para mostrar que butaca o asiento está siendo ocupado en ese momento y en el *display* LCD se mostrará el estado total de asientos que están libres u ocupados dentro de la unidad de transporte.

```
void leer_FC(){
  lcd.clear();
  lcd.setCursor(4,0);
  lcd.print("ASIENTOS");
  lcd.setCursor(0,1);
  lcd.print("LIB:");
  lcd.setCursor(4,1);
  lcd.print(libres);
  lcd.setCursor(8,1);
  lcd.print("OCU:");
  lcd.setCursor(13,1);
  lcd.print(n);
  delay(100);
```

Figura 4.12 Líneas de código para mostrar información el *display* LCD

Fuente: Elaborado por el autor

Se deberán configurar los parámetros iniciales del LDC y mensajes a mostrar cuando este se encuentre en modo *standby*, como se puede apreciar en la figura anterior, después se deberá proceder a comparar cuando el final de carrera se acciona e inmediatamente se enviará a través del puerto serial un mensaje indicando que el pasajero se encuentra sentado, este mensaje también será enviado al usuario o socio de la unidad mediante un SMS.

```
if(val_FCA1==LOW){
  digitalWrite(led_rojo_A1,HIGH);
  digitalWrite(led_verde_A1,LOW);
  if (cont1==0 && aux==1){
  cont2=0;
  Serial.print("PASAJERO 1         ");
  enviar_mensaje_plsentado();}
  cont1++;}
else{
  digitalWrite(led_rojo_A1,LOW);
  digitalWrite(led_verde_A1,HIGH);
  if (cont2==0 && aux==1){
  cont1=0;
  Serial.print("PASAJERO 1         ");
  enviar_mensaje_plparado();}
  cont2++;}
```

Figura 4.13 Líneas de código para validar la butaca o asiento

Fuente: Elaborado por el autor

Como se presenta en la figura anterior adicionalmente al enviar el mensaje a través del puerto serial hacia la base datos y SMS hacia el usuario, se enviará una señal hacia un LED ubicado en el panel de pasajeros el cual indica que butaca o asiento se ha ocupado, donde el LED de color verde indica que el asiento está disponible, y el LED rojo indica que el asiento se encuentra ocupado.

4.1.9 Servidor LAMP dentro del microcomputador Raspberry PI.

Luego de haber verificado que el Arduino pueda detectar cuando un interruptor final de carrera se haya accionado, el sensor magnético esté funcionando de acuerdo a lo programado, el módulo laser detecte cuando un pasajero ingrese o salga de la unidad y que el módulo SIM808 GSM/GPS se encuentre operativo, se deberá proceder con la instalación de un servidor web para ello usaremos *software* que se acople a el sistema operativo que maneja Raspberry PI.

Antes de instalar un servidor web como recomendación se debe actualizar el sistema operativo de la Raspberry PI, con el siguiente comando se ejecutará y verificará si existe alguna actualización disponible del sistema operativo.

```
pi@raspberrypi:~ $ sudo rpi-update
```

Figura 4.14 Comando para verificar actualizaciones

Fuente: Elaborado por el autor

Una vez que se haya verificado si existe alguna actualización disponible se procederá a realizar un reinicio del sistema, después se procederá con la actualización del sistema con los siguientes comandos.

```
pi@raspberrypi:~ $ sudo apt-get update
pi@raspberrypi:~ $ sudo apt-get upgrade
```

Figura 4.15 Comando para actualizar Raspberry PI

Fuente: Elaborado por el autor

Luego de ello se deberá instalar varias herramientas en Raspberry para el funcionamiento de la interfaz web y la creación de la base de datos.

4.1.10 Instalación de Apache como servidor.

El servidor Apache es el encargado de escuchar las peticiones que realicen los usuarios a la página web de la base de datos que se va a implementar, para poder instalar dicho servidor se debe ejecutar el siguiente comando.

```
pi@raspberrypi:~ $ sudo apt install apache2
```

Figura 4.16 Comando para instalar Apache

Fuente: Elaborado por el autor

Una vez que se haya completado la instalación se podrá verificar el funcionamiento con el siguiente comando que se muestra en la figura a continuación, y se desplegará en la tercera línea el estado en el que se encuentra.

```
pi@raspberrypi:~ $ sudo service apache2 status
pi@raspberrypi:~ $ sudo service apache2 status
● apache2.service - The Apache HTTP Server
   Loaded: loaded (/lib/systemd/system/apache2.s
   Active: active (running) since Mon 2019-06-03
  Process: 468 ExecStart=/usr/sbin/apachectl sta
 Main PID: 614 (apache2)
    Tasks: 8 (limit: 4915)
```

Figura 4.17 Comando para verificar estatus de Apache

Fuente: Elaborado por el autor

4.1.11 Instalación de PHP.

Una vez verificado que se haya instalado el servidor Apache se deberá proceder con la instalación de PHP mediante el siguiente comando.

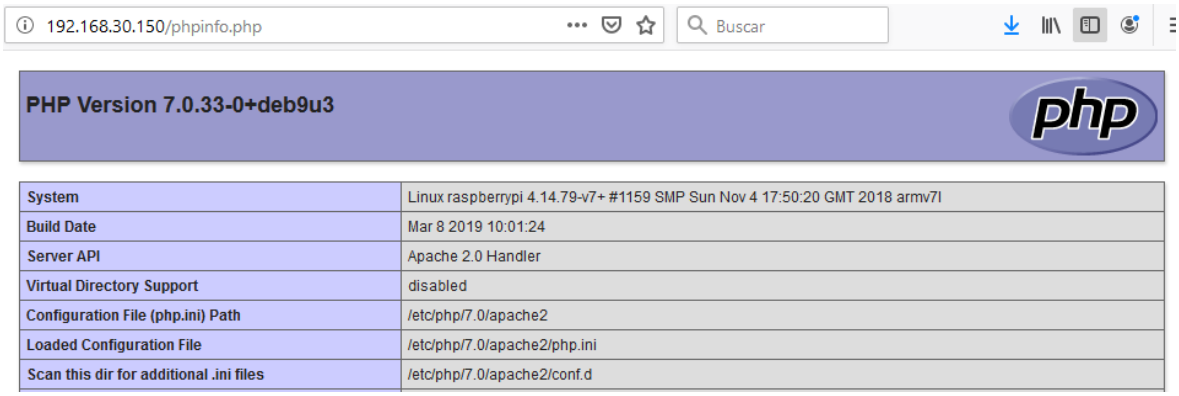
```
pi@raspberrypi:~ $ sudo apt install -t stretch -y php7.0 libapache2-mod-php7.0 php7.0-mysql
```

Figura 4.18 Comando para instalar PHP

Fuente: Elaborado por el autor

A través de PHP se deberá visualizar el todo contenido y también la posibilidad de crear páginas web dinámicas que interactúen con el usuario, devolviendo así datos en tiempo real de lo que está aconteciendo en ese momento.

Para poder verificar que PHP se haya instalado de forma correcta, se deberá abrir un navegador web y se digitara en la URL los siguiente “http://192.168.30.150/phpinfo.php”, en el cual se mostrará la siguiente página que se muestra en la siguiente figura.



System	Linux raspberrypi 4.14.79-v7+ #1159 SMP Sun Nov 4 17:50:20 GMT 2018 armv7l
Build Date	Mar 8 2019 10:01:24
Server API	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php/7.0/apache2
Loaded Configuration File	/etc/php/7.0/apache2/php.ini
Scan this dir for additional .ini files	/etc/php/7.0/apache2/conf.d

Figura 4.19 Validación del correcto funcionamiento de PHP

Fuente: Elaborado por el autor

4.1.12 Instalación de MySQL.

Para continuar se debe proceder a instalar MySQL, para administrar y almacenar los datos enviados por los finales de carrera, el sensor magnético, módulo GSM/GPS y el módulo laser, con el siguiente comando se procederá con la instalación.

```
pi@raspberrypi:~ $ sudo apt-get install mysql-server
```

Figura 4.20 Comando para instalar MySQL

Fuente: Elaborado por el autor

4.1.13 Instalación de PhpMyAdmin.

Con la instalación de PhpMyAdmin permitirá la administración de la base de datos construida en el gestor de MySQL mediante el uso de cualquier *browser*, es decir, se podrá modificar la base de datos de forma gráfica y más amigable, mostrar los resultados almacenados, gestionar usuarios de MySQL, etc., para ello ejecutaremos el siguiente comando.

```
pi@raspberrypi:~ $ sudo apt-get install phpmyadmin
```

Figura 4.21 Comando para instalar PhpMyAdmin

Fuente: Elaborado por el autor

Para completar con el proceso se debe editar el archivo de configuración de Apache para ello se ejecuta el siguiente comando, e inmediatamente se deberá insertar la siguiente línea de código al final del texto “Include /etc/phpmyadmin/apache.conf”, como se muestra en la Figura 4.23 después se deberá guardar los cambios realizados.

```
pi@raspberrypi:~ $ sudo nano /etc/apache2/apache2.conf
```

Figura 4.22 Comando para editar archivo apache2.conf

Fuente: Elaborado por el autor

```
      Fichero: /etc/apache2/apache2.conf

# Include generic snippets of statements
IncludeOptional conf-enabled/*.conf

# Include the virtual host configurations:
IncludeOptional sites-enabled/*.conf

# vim: syntax=apache ts=4 sw=4 sts=4 sr noet

Include /etc/phpmyadmin/apache.conf
```

Figura 4.23 Insertando línea dentro el archivo apache2.conf

Fuente: Elaborado por el autor

Para comprobar que se haya instalado de forma correcta se deberá verificar a través de un navegador web con la siguiente URL “http://192.168.30.150/phpmyadmin/”.

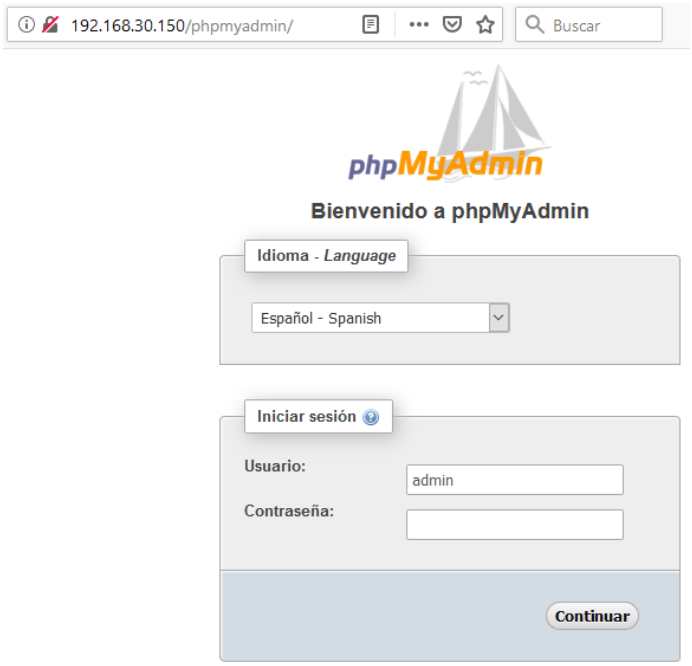


Figura 4.24 Visualización de PhpMyAdmin vía browser

Fuente: Elaborado por el autor

A continuación, se deberá proceder a crear un usuario con privilegios full, para administrar la base de datos, esto se lo realizará dentro de la interfaz MySQL, con el siguiente comando, posteriormente se deberá dar permisos para acceder al servidor de MySQL

```
MariaDB [(none)]> CREATE USER 'admin' IDENTIFIED BY 'root$$2018';
MariaDB [(none)]> GRANT USAGE ON *.* TO 'admin'@localhost IDENTIFIED BY 'root$$2018';
```

Figura 4.25 Creación y asignación de permisos a usuario admin

Fuente: Elaborado por el autor

Se comprobará el acceso al servidor MySQL través de un navegador web, con las credenciales creadas anteriormente a continuación, se desplegará la siguiente página ingresando a la siguiente URL http://192.168.30.150/phpmyadmin/

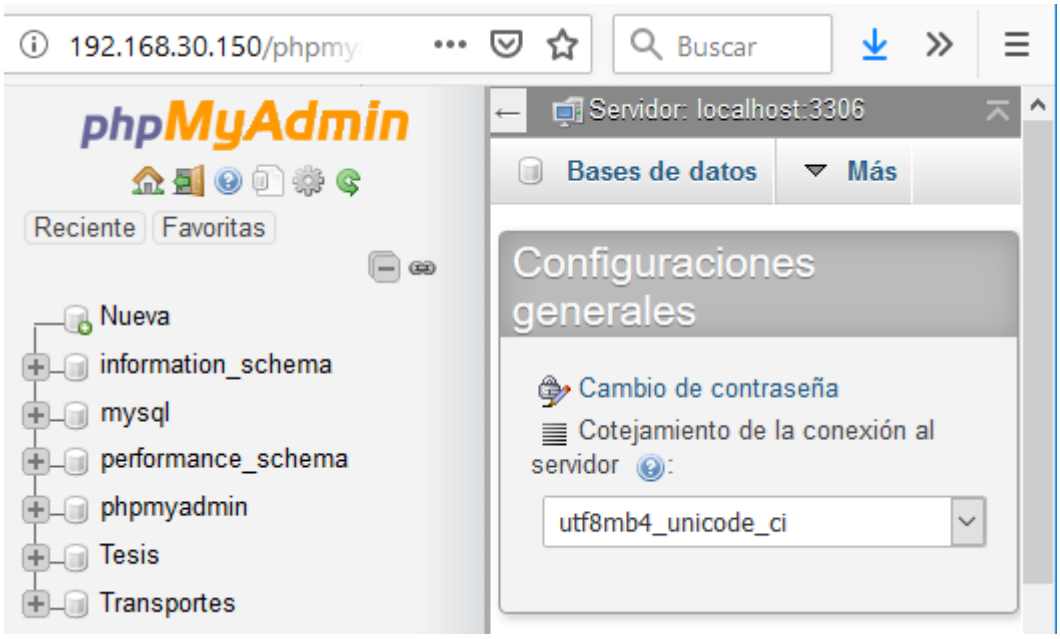


Figura 4.26 Interfaz del servidor de base de datos

Fuente: Elaborado por el autor

La dirección IP que se muestra en la dirección URL es la que asignó de forma estática al prototipo para realizar pruebas de funcionamiento, esta a su vez se puede acceder vía remota dentro de la red interna a través de SSH.

Una vez verificado el acceso al servidor MySQL se procederá a desarrollar el código de programación que permitirá la comunicación entre Arduino y Raspberry PI, para de esta manera almacenar los datos procesados por el microcontrolador.

4.1.14 Diseño de la base de la base de datos.

Se procede a crear la base de datos se usará la herramienta de diseño y construcción MySQL *Workbench*, el entorno de esta herramienta es muy amigable y fácil de usar, la base de datos contendrá 8 tablas relacionadas entre sí para que de esta manera se pueda seleccionar y encontrar de forma más rápida eficaz la información, como se muestra en la Figura 4.27 se estable las relaciones de la siguiente manera:

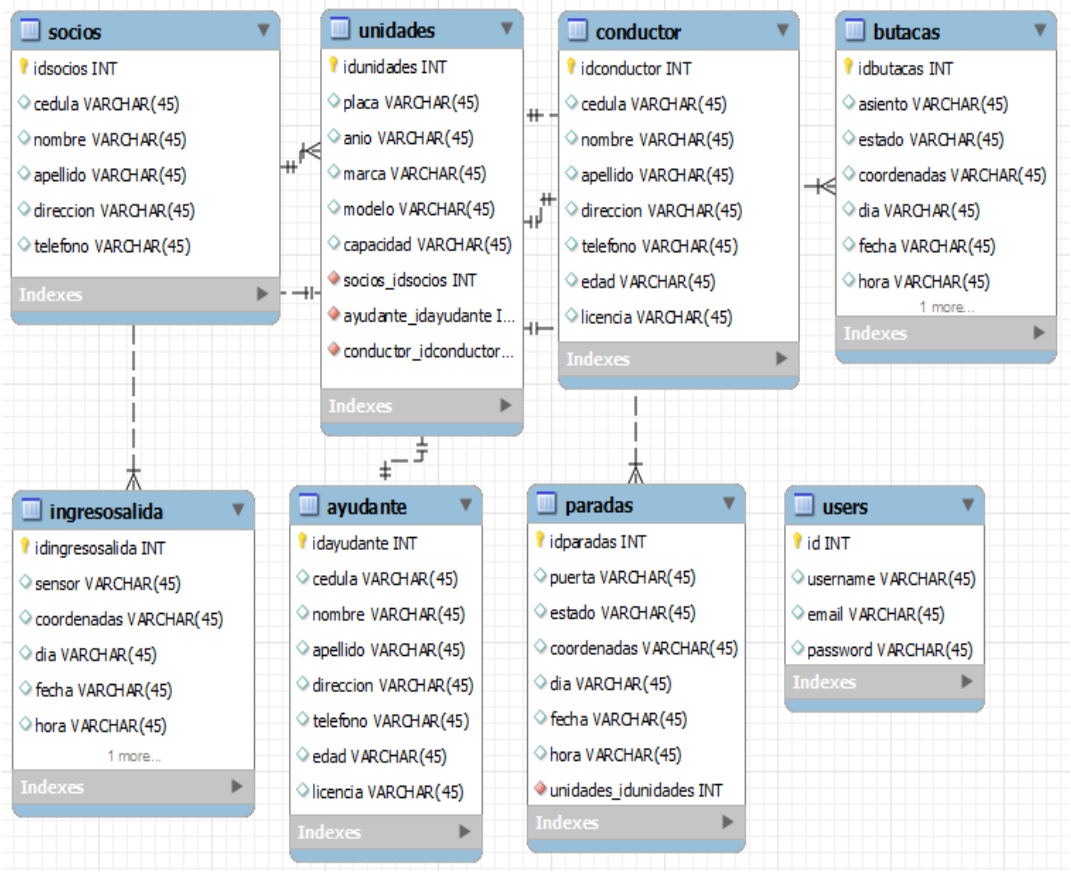


Figura 4.27 Relaciones y tablas de la base de datos

Fuente: Elaborado por el autor

La tabla socios y tabla unidades está relacionada de uno a muchos

La tabla conductor y tabla unidades está relacionada de uno a muchos

La tabla ayudante y tabla unidades está relacionada de uno a muchos

La tabla unidad y tabla paradas está relacionada de uno a muchos

La tabla unidad y tabla butacas está relacionada de uno a muchos

La tabla unidad y tabla ingresosalida está relacionada de uno a muchos

A continuación, se deberá guardar la base datos creada para ejemplos de prototipo se lo nombrará como “Transportes”, después se importa el código fuente de la base de datos para insertarlo en el servidor de MySQL, ingresando a través de un navegador web.

4.1.15 Comunicación de Arduino Mega y Raspberry.

Para realizar el enlace entre Raspberry y Arduino Mega, se deberá cargar un script creado en Python, en el cual dentro del archivo se desarrollará el código de programación que hace posible dicha comunicación.

Para crear el archivo Python dentro de Raspberry ejecutaremos el siguiente comando, el cual permitirá la creación del archivo con la extensión .py

```
pi@raspberrypi:~ $ sudo nano RaspDuino/RaspDuino.py
```

Figura 4.28 Comando para crear archivo RaspDuino.py

Fuente: Elaborado por el autor

Después se deberá importar las siguientes librerías que permitirá con el desarrollo del programa, como son la de comunicación serial, comunicación con la base de datos MySQL, la de hora y fecha, etc., en seguida se declarará una variable el cual recibirá los datos que lleguen a través del puerto serial enviados desde el microcontrolador Arduino Mega.

```
  GNU nano 2.7.4                Fichero: RaspDuino/RaspDuino.py

import glob
import time
import commands
import os
import socket
import serial
import mysql.connector

arduino = serial.Serial('/dev/ttyACM0', baudrate=9600)
```

Figura 4.29 Líneas de código para importar librerías

Fuente: Elaborado por el autor

Como se observa en la figura anterior se importa las librerías más importantes que hacen posible la comunicación entre Arduino Mega y Raspberry PI, de la misma manera se asigna una variable para recibir los datos en texto plano.

Enseguida se crea un ciclo, se asigna una variable local y se fragmenta los datos recibidos para poder ser interpretados y enviados a la base de datos.

```
  GNU nano 2.7.4                    Fichero: RaspDuino/RaspDuino.py


while True:
        line = arduino.readline().strip()
        g = line[0:6]
        a = line[0:18]
        b = line[19:41]
        c = line[42:52]
        d = time.strftime("%A")
        e = time.strftime("%d %b %Y")
        f = time.strftime("%X")
```

Figura 4.30 Líneas de código para leer datos a través del puerto serial

Fuente: Elaborado por el autor

Adicionalmente se usará el reloj interno del microcomputador para asignar la fecha y hora a cada evento que ocurra dentro de la unidad, estos valores se encuentran almacenados en las variables locales d, e y f, como se muestra en la figura anterior.

```
w = "PASAJE"
if (g == w):
        dato = {'user':'admin','password':'root$$2018','database':'Transportes','host':'localhost'}
        conexion = mysql.connector.connect(** dato)
        cursor = conexion.cursor()
        valores = ("""INSERT INTO butacas (asiento,estado,coordenadas,dia,fecha,hora)
                    VALUES (%s,%s,%s,%s,%s,%s)""",(a,c,b,d,e,f))
        try:
                cursor.execute(*valores)
                conexion.commit()
        except:
        conexion.close()
```

Figura 4.31 Líneas de código para filtrar datos de butacas

Fuente: Elaborado por el autor

Después se asigna una constante cuyo propósito es la de comparar el dato recibido con la cadena de datos proporcionada por el microcontrolador, si esto se cumple se conectará a la base de datos a través de las credenciales creadas anteriormente y guardará el valor en la tabla correspondiente anteriormente creada, en la figura anterior se observa las líneas de código que hacen que los datos sean insertados dentro de la base de datos.

De la misma manera se deberá proseguir para recibir y guardar los demás datos que sean necesario almacenarlos dentro de sus tablas correspondientes.

4.1.16 Diseño de la interfaz web.

Después de concluir con el diseño de la base de datos, se deberá diseñar la interfaz de visualización para el usuario o socio de la compañía, para ello se debe crear varios registros con la extensión. php para las distintas páginas del prototipo cuyo propósito es la de enlazarse con la base de datos creada anteriormente, para escribir el código fuente de cada página se lo podrá realizar en un editor de texto como el bloc de notas, Notepad++, Sublime text etc., se deberán guardar en la siguiente dirección /var/www/html/ del microcomputador Raspberry PI, a continuación, se presentan las paginas creadas:

index.php: Contiene la página principal del prototipo.

register.php: Contiene la página para registrar un nuevo usuario

errors.php: Contiene la página cuando se produce un error de inicio de sesión.

butacas.php: Contiene la página del estado de todas las butacas de la unidad.

paradas.php: Contiene la página del estado de las paradas realizadas de la unidad.

ingresosalida.php: Contiene la página del estado del ingreso y salida de pasajeros.

login.php: Contiene la página para iniciar de sesión.

server.php: Contiene el código fuente que permite la interacción entre la página web y la base de datos.

Una vez creadas estas páginas se las deberá subir al servidor web del microcomputador en la carpeta antes mencionada, para ello se crea una carpeta con el nombre de *registration* dentro de la carpeta de html en la cual se insertan las páginas desarrolladas.

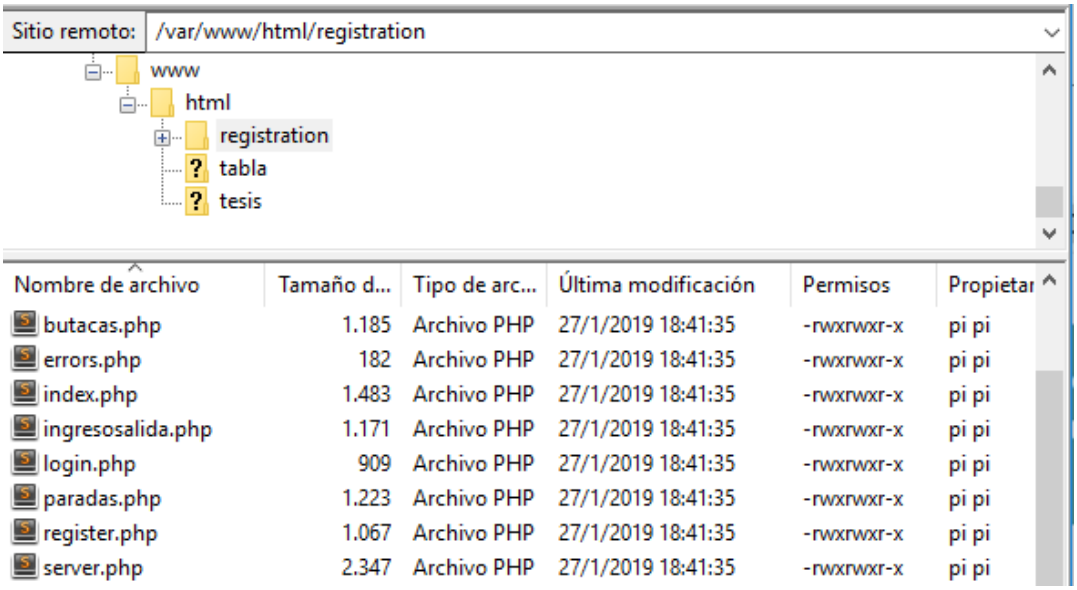


Figura 4.32 Páginas creadas para la comunicación con la base de datos

Fuente: Elaborado por el autor

Para verificar que está funcionando correctamente se deberá visualizar la siguiente página que se muestra en la Figura 4.33. Para ello se abrirá un navegador y se ingresa a la siguiente URL http://192.168.30.150/registration/login.php

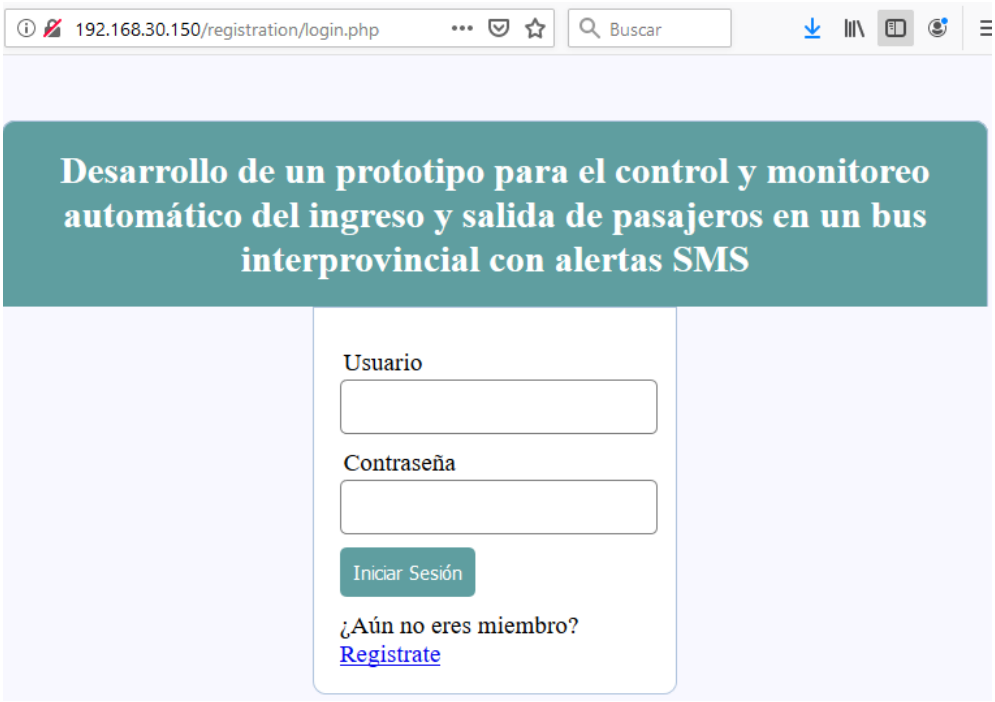


Figura 4.33 Interfaz de inicio del prototipo

Fuente: Elaborado por el autor

4.1.17 Diseño electrónico.

El diseño electrónico se lo presentará en diferentes partes, se muestra el diseño electrónico y esquema de conexión con el módulo SIM808 GSM/GPS, los finales de carrera, el modulo laser, el *display* LCD, el panel LED de pasajeros y alimentación del circuito que se conectarán al microcontrolador.

4.1.18 Diagrama electrónico de conexión entre Arduino y el módulo SIM808 GSM/GPS

De acuerdo a la programación realizada los puertos que se deben conectar a la transmisión (TX) y la recepción (RX) del módulo SIM808 son el puerto 10 puerto 11 respectivamente en la siguiente figura se puede apreciar dicha conexión.

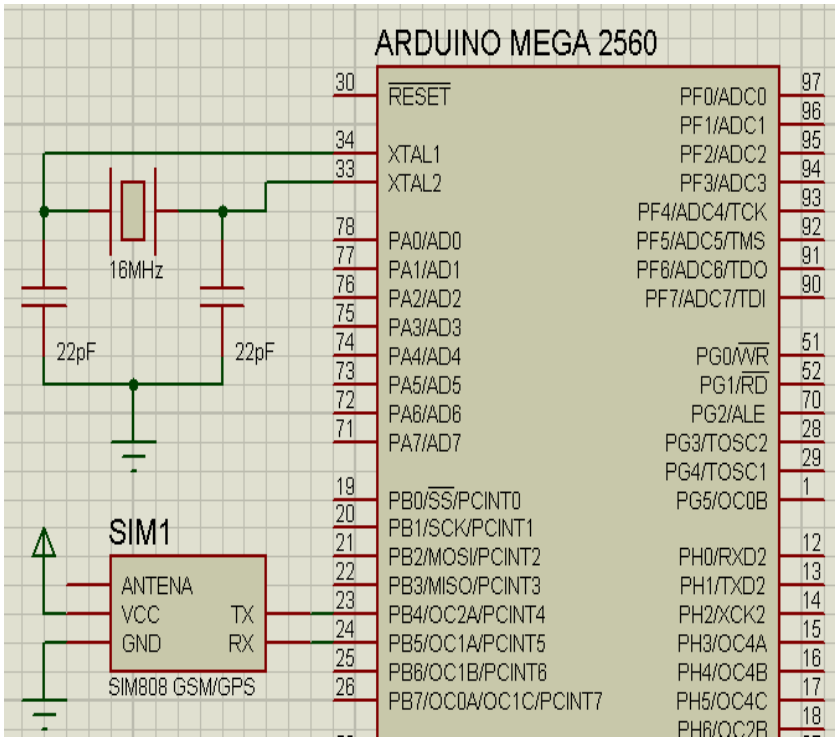


Figura 4.34 Esquema de conexiones del módulo SIM808 GSM/GPS

Fuente: Elaborado por el autor

4.1.19 Diagrama electrónico de conexión entre Arduino y finales de carrera.

De acuerdo a la programación realizada los puertos que se deben conectar a cada final de carrera serán desde el puerto 22 hasta el puerto 29, tomando en cuenta que FC1 se conecta al puerto 22, FC2 al puerto 23 y así respectivamente en la siguiente figura se puede apreciar dicha conexión.

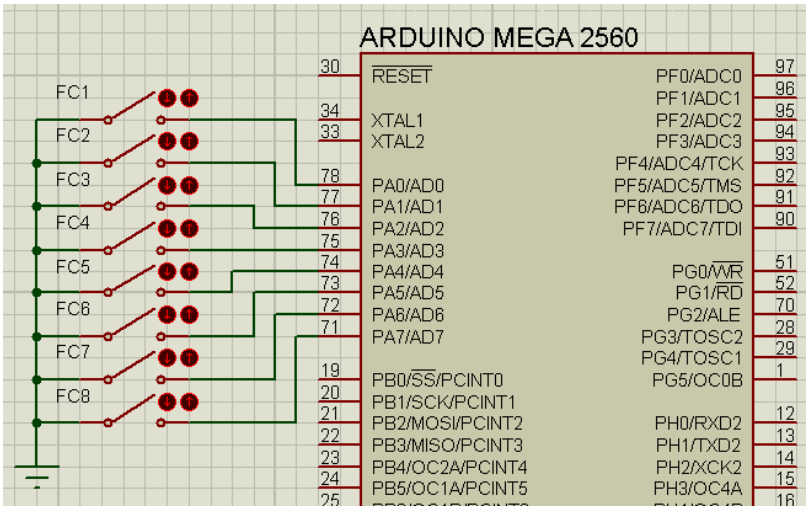


Figura 4.35 Esquema de conexión de los finales de carrera

Fuente: Elaborado por el autor.

4.1.20 Diagrama electrónico de conexión entre Arduino y el panel LED de pasajeros.

De acuerdo a la programación realizada los puertos que se deberá conectar a cada diodo emisor de luz (verde) serán desde el puerto 30 hasta el puerto 37, este LED indicará que la butaca no está siendo ocupada y está disponible, de la misma manera se deberá conectar a cada diodo emisor de luz (rojo) desde el puerto 38 hasta el puerto 45, este LED indicará que la butaca está siendo ocupada y no está disponible, en las siguientes figuras se puede apreciar dichas conexiones hacia el microcontrolador.

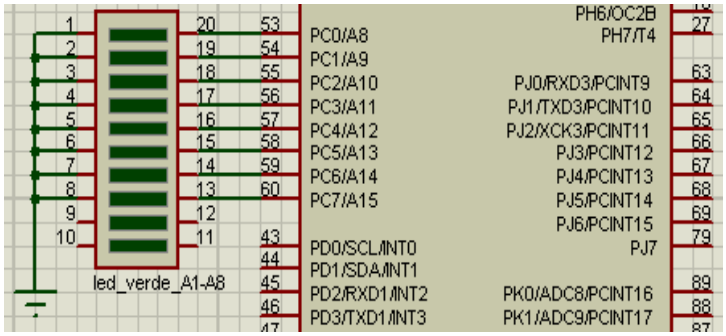


Figura 4.36 Esquema de conexión de los diodos LED verdes

Fuente: Elaborado por el autor

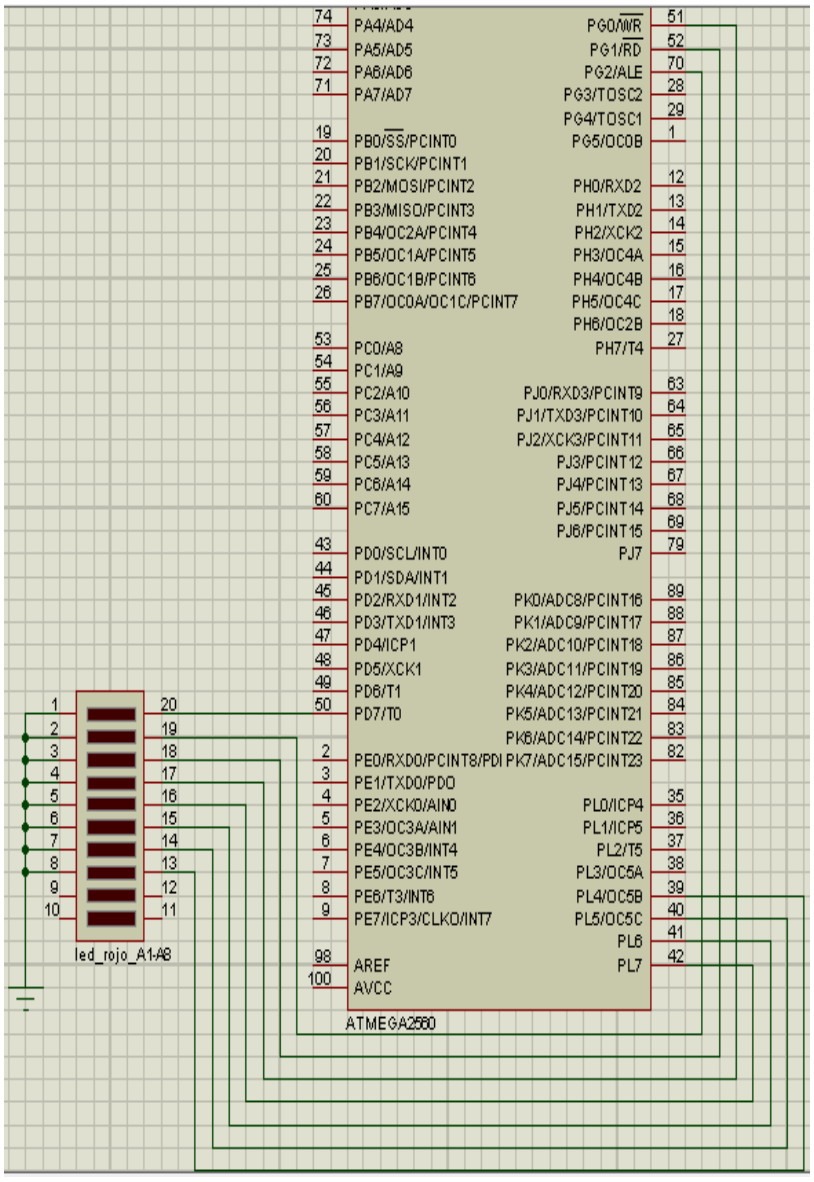


Figura 4.37 Esquema de conexión de diodos LED rojos

Fuente: Elaborado por el autor

4.1.21 Diagrama electrónico de conexión entre Arduino y *display* LCD.

De acuerdo a la programación realizada los puertos de conexión que se deben conectar al puerto SDA y puerto SCL del módulo I2C serán el puerto 20 y puerto 21 del microcontrolador respectivamente en la siguiente figura se puede apreciar dicha conexión.

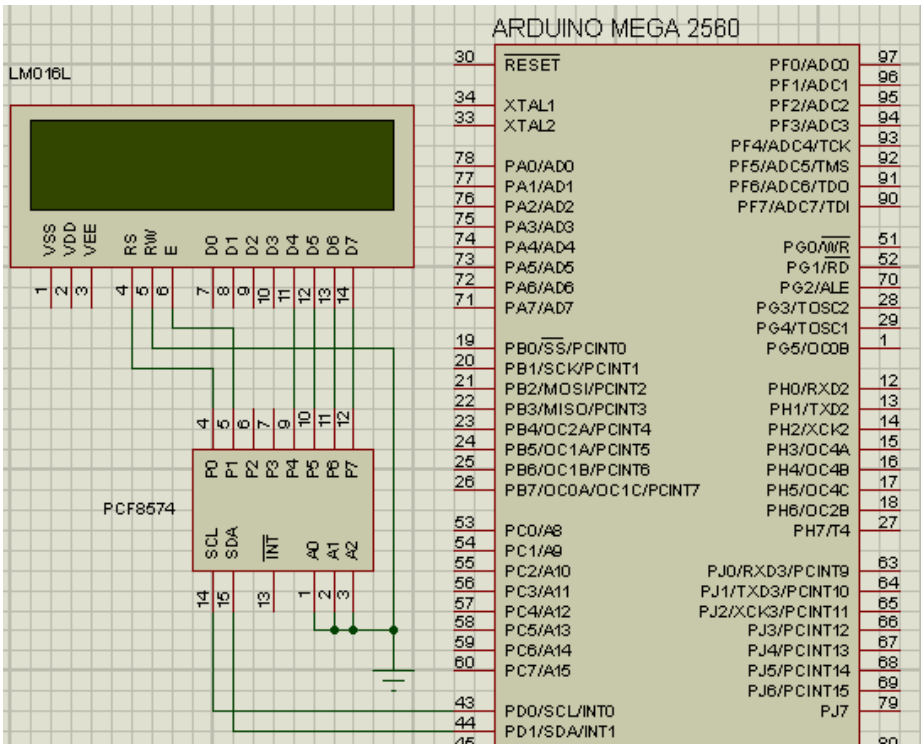


Figura 4.38 Esquema de conexión del *display* LCD

Fuente: Elaborado por el autor

4.1.22 Diagrama electrónico de conexión entre Arduino y sensor magnético.

De acuerdo a la programación realizada el puerto 46 se usará para detectar la señal que envié el sensor magnético al microcontrolador en la siguiente figura se puede apreciar dicha conexión.

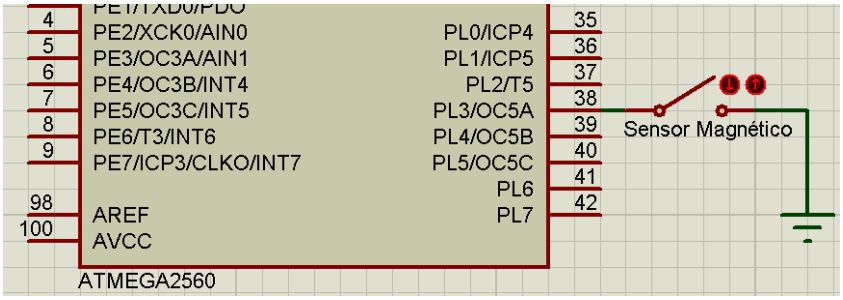


Figura 4.39 Esquema de conexión del sensor magnético

Fuente: Elaborado por el autor

4.1.23 Diagrama electrónico de conexión entre Arduino y módulo laser.

De acuerdo a la programación realizada los puertos de conexión para el receptor del módulo laser (LDR) serán el puerto 47 y el puerto 48 en esta ocasión se usará dos módulos laser para cumplir con el propósito de verificar el ingreso y salida de pasajeros. En la siguiente figura se puede apreciar dicha conexión

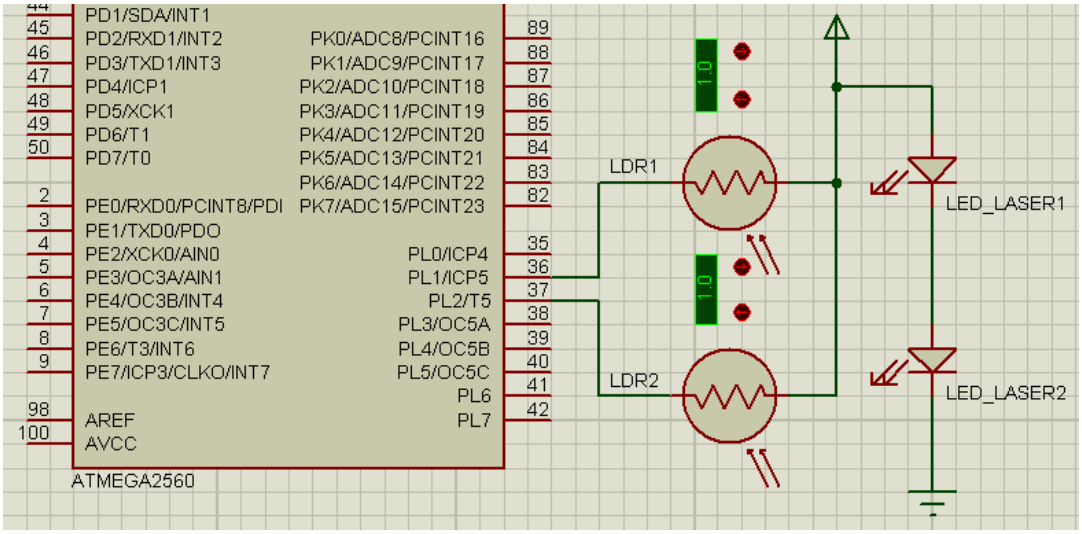


Figura 4.40 Esquema de conexión de módulo laser

Fuente: Elaborado por el autor

.

4.1.24 Diagrama electrónico de conexión entre Arduino y la alimentación del circuito.

Para tener un sistema redundante se instaló un banco de baterías que suministrarán, a su salida un valor de 5 voltios (5V) y 2 amperios (2A) al prototipo necesarios para obtener un óptimo funcionamiento, el banco de baterías será alimentado desde la batería que dispone la unidad de transporte que es de 12 voltios (12V) por lo general.

De acuerdo a la hoja de especificaciones del fabricante del banco de baterías la alimentación del mismo debe ser de 5 voltios (5V) por que es necesario colocar un regulador de voltaje, para ello se usará el circuito integrado LM2596. En la siguiente figura se puede apreciar dicha conexión entre los diferentes elementos electrónicos.

El banco de baterías tendrá una duración de 8 horas según el análisis que se realizó posteriormente, si la unidad de transporte llegara a quedarse sin energía, por lo que el prototipo podrá seguir operando normalmente.

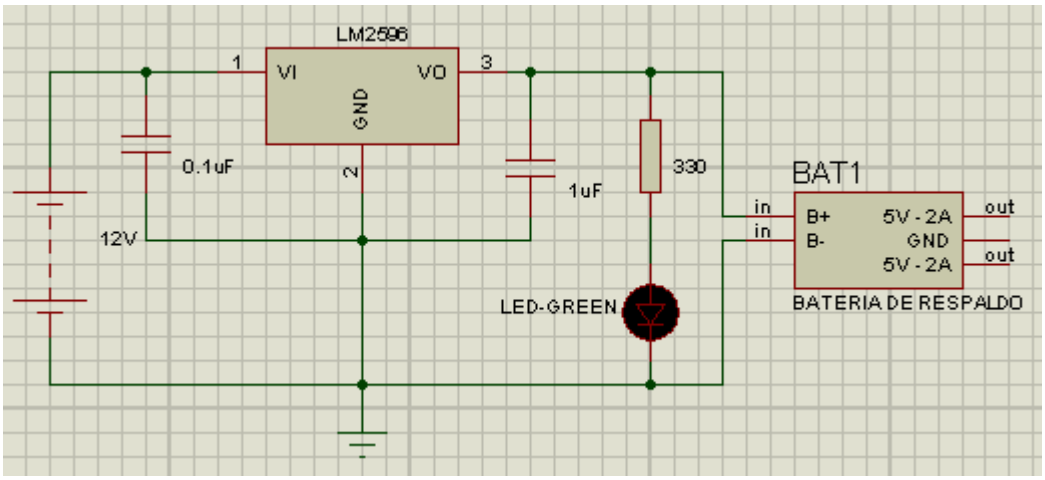


Figura 4.41 Esquema de conexión de regulador de voltaje

Fuente: Elaborado por el autor

4.1.25 Dimensionamiento de fuente de alimentación del circuito.

Para realizar el dimensionamiento de alimentación electrónica adecuada, es indispensable elegir una la fuente de alimentación adecuada, que se ajuste al consumo de corriente que tiene cada uno de los dispositivos o elementos electrónicos que constituye el prototipo, para llevar a cabo este proceso se debe realizar un análisis teórico con las especificaciones que dicta el fabricante dentro de hoja de especificaciones que dicta el fabricante de cada uno de los elementos que conforman el prototipo y de manera práctica mediante el uso de una herramienta electrónica como lo es el multímetro.

Dentro de la tabla 4.1, se detallan la cantidad de corriente teórica y práctica que consumen los diferentes elementos electrónicos a usarse.

Tabla 4.1 Consumo de corriente teórica y práctica

	Consumo Teórico	**Consumo Medido**	**Voltaje de operación**
Arduino Mega 2560	93 mA	98 mA	5V
Raspberry PI	310 mA	320 mA	5V
Módulo Laser (TX)	40 mA	43 mA	5V
***Display* LCD**	25 mA	27 mA	5V
LED verde x 8	160 mA	160 mA	5V
LED rojo x 8	120 mA	120 mA	5V
Total	748 mA	768 mA	No aplica

Fuente: Elaborado por el autor

Una vez calculado el consumo de corriente total del circuito se debe buscar una fuente o batería que suministre la cantidad necesaria de corriente que se necesita para el correcto funcionamiento del prototipo.

Dentro del país existe una fuente de baterías de 5V a 6000 mAh el cual contiene 5 celdas de litio, con esta batería se podrá alimentar a todo el circuito ya que se encuentra dentro del consumo adecuado de corriente.

$$P = 6000\text{mA x } 5\text{V} = 30\text{ W}$$

Ecuación 4.1 Ley de ohm

Fuente: (Boylestad, 2009)

Tiempo promedio de alimentación del prototipo.

$$Tiempo\ de\ funcionamiento = \frac{6000\text{mAh}}{768\text{ mA}} = 7{,}8\ Horas$$

Ecuación 4.2 Tiempo promedio de funcionamiento

Fuente: Elaborado por el autor

Según el resultado anterior obtenido es aproximadamente de 7 horas con 45 minutos, este tiempo es el que estará encendido mediante el uso de las baterías en el momento que la unidad de transporte se quede sin energía.

Para determinar el tiempo que tarda en cargar una batería a través de un adaptador de corriente de 5V y 2000 mA, para ello circuito integrado LM2596 se usa como el regulador de voltaje cumple con este requerimiento a continuación, se determina mediante la siguiente formula.

$$Tiempo\ de\ carga\ de\ batería = \frac{6000mAh}{2000mA} = 3\ horas$$

Ecuación 4.3 Tiempo de carga de batería

Fuente: Elaborado por el autor

4.2 Implementación.

Para la implementación del prototipo se realizará el ensamblado de todos elementos que conforman el proyecto como son: la implementación de el interruptor final de carrera en el cinturón de seguridad, implementación del módulo SIM808 GSM/GPS, implementación del panel de pasajeros, implementación del sensor magnético y comunicaciones, implementación del módulo laser y la conexión de Raspberry PI con Arduino Mega 2560.

4.2.1 Instalación del interruptor final de carrera en el cinturón de seguridad.

Para realizar la instalación de los finales de carrera en cada cinturón de seguridad se debe modificar dicho cinturón e insertar el final de carrera para luego ser conectado al microcontrolador.

En la Figura 4.42, se puede observar el cinturón de seguridad sin la cubierta posterior que el lugar se instalará el interruptor final de carrera.

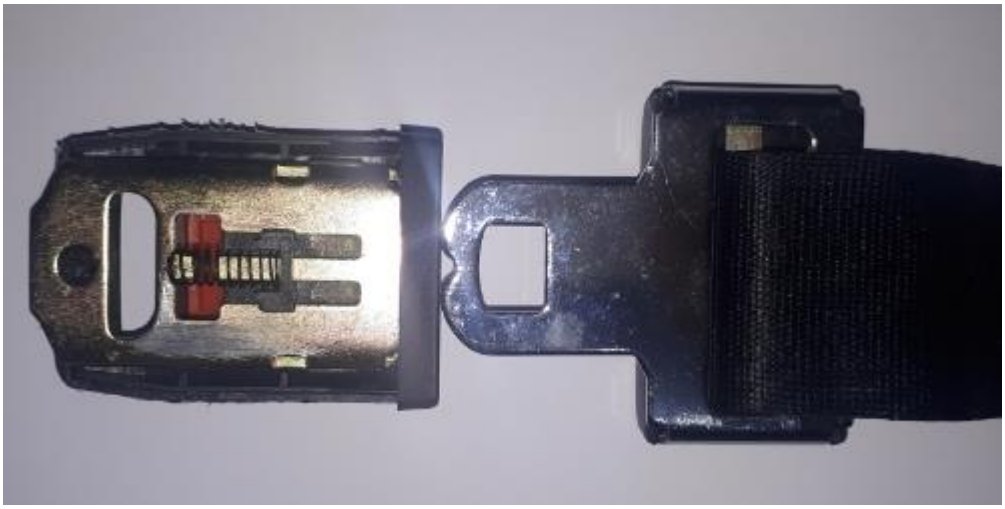


Figura 4.42 Cinturón de seguridad desarmado

Fuente: Elaborado por el autor

Para realizar este procedimiento primero se debe soldar un par cables en el terminal normalmente abierto y la salida común como se muestra en la Figura 4.43 por lo que cada vez que se accione el final de carrera enviara una señal hacia el microcontrolador.

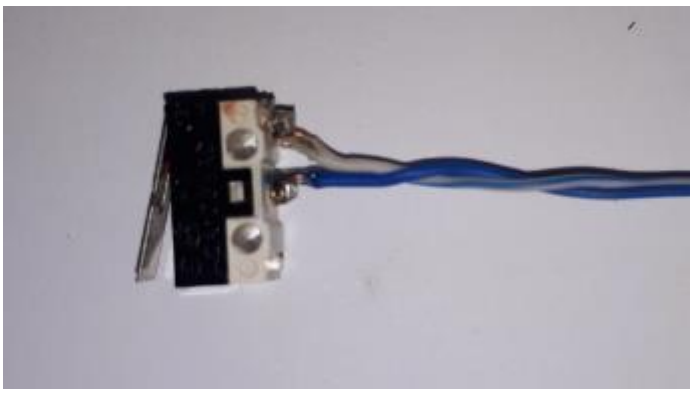


Figura 4.43 Soldando final de carrera

Fuente: Elaborado por el autor

Después se debe colocar el interruptor final de carrera dentro del cinturón de seguridad ya que cada vez que el usuario se abroche el cinturón este accionará el final de carrera, como se muestra en la Figura 4.44 se visualiza la instalación ya realizada, de la misma manera se deberá proseguir con los 7 interruptores restantes con sus respectivos cinturones de seguridad.

Figura 4.44 Instalación de final de carrera en cinturón de seguridad

Fuente: Elaborado por el autor

4.2.2 Implementación de los interruptores finales de carrera.

Después de haber concretado con la instalación de los finales de carrera en cada cinturón de seguridad se deberá conectar hacia los puertos previamente configurados en la plataforma de Arduino IDE. Een la Figura 4.45 se puede apreciar la instalación de los diferentes finales de carrera hacia el Arduino Mega 2560.

Figura 4.45 Instalación de finales de carrera en Arduino

Fuente: Elaborado por el autor

4.2.3 Instalación de LED's en el panel de pasajeros.

En las Figura 4.46, Figura 4.47 y Figura 4.48, se puede apreciar el panel de pasajeros, la implementación de los diferentes diodos emisores de luz y su conexión hacia el microcontrolador.

Figura 4.46 Panel de pasajeros

Fuente: Elaborado por el autor

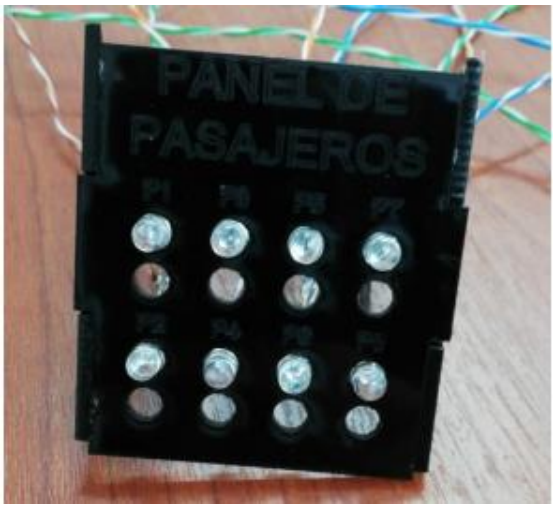


Figura 4.47 Instalación de diodos LED en el panel de pasajeros

Fuente: Elaborado por el autor.

Figura 4.48 Instalación de diodos LED en Arduino

Fuente: Elaborado por el autor

4.2.4 Instalación de módulo laser.

En la Figura 4.49, se puede visualizar la instalación de cada módulo laser, esta implementación se la debe realizar en la puerta más cercana al pasillo de la maqueta del bus interprovincial, para de esta manera simular el ingreso y salida de pasajeros.

Figura 4.49 Instalación de módulo laser (transmisión y recepción)

Fuente: Elaborado por el autor

4.2.5 Instalación del *display* LCD

En la Figura 4.50, se puede visualizar la instalación del *display* LCD, en ella se mostrará la cantidad de asientos disponibles y la cantidad de asientos ocupados.

Figura 4.50 Instalación de *display* LCD

Fuente: Elaborado por el autor.

4.2.6 Instalación de módulo SIM808 GSM/GPS.

En la Figura 4.51, se puede visualizar la instalación de la antena GSM y antena GPS para el módulo, la alimentación vendrá de uno de los puertos de 5V a 2A para un óptimo funcionamiento.

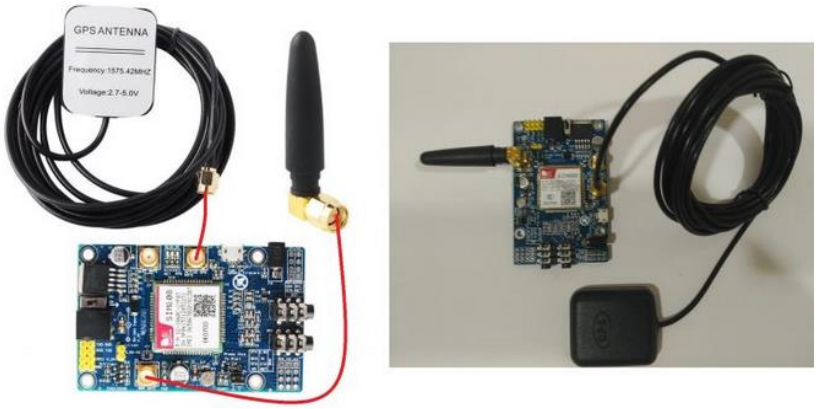


Figura 4.51 Instalación de módulo SIM808 GSM/GPS

Fuente: Elaborado por el autor

4.2.7 Instalación del microcontrolador con Raspberry PI.

En la Figura 4.52, se puede visualizar la instalación del Raspberry PI y el microcontrolador Arduino Mega 2560, la alimentación del microcontrolador la proveerá la Raspberry PI y la alimentación del microcomputador vendrá de uno de los puertos de 5V a 2A para un óptimo funcionamiento.

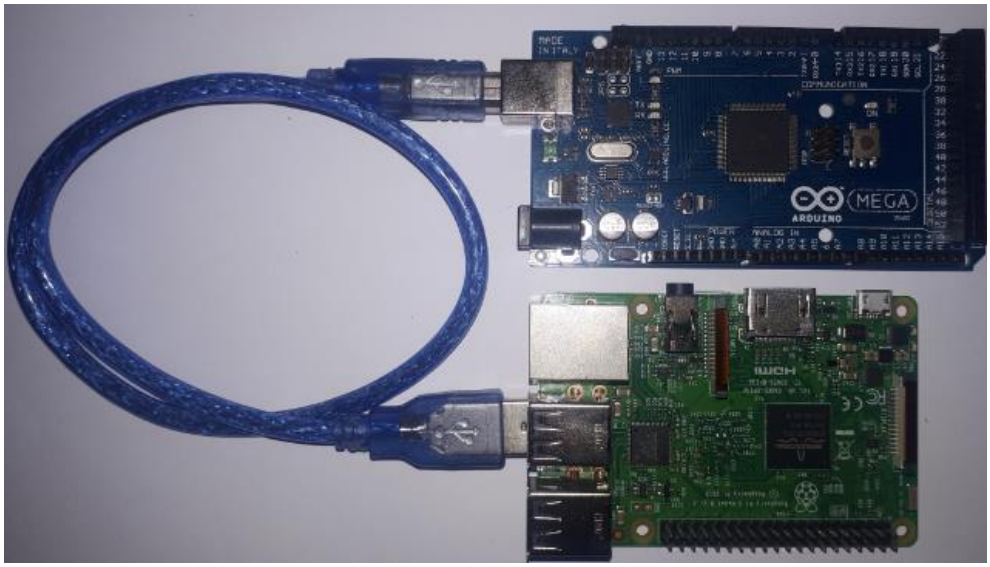


Figura 4.52 Conexión serial entre Arduino y Raspberry PI

Fuente: Elaborado por el autor

4.2.8 Verificación del armado.

En la Figura 4.53, se puede verificar el armado completo de todos los elementos que intervienen en el prototipo para el control y monitoreo automático del ingreso y salida de pasajeros en un bus interprovincial con alertas SMS.

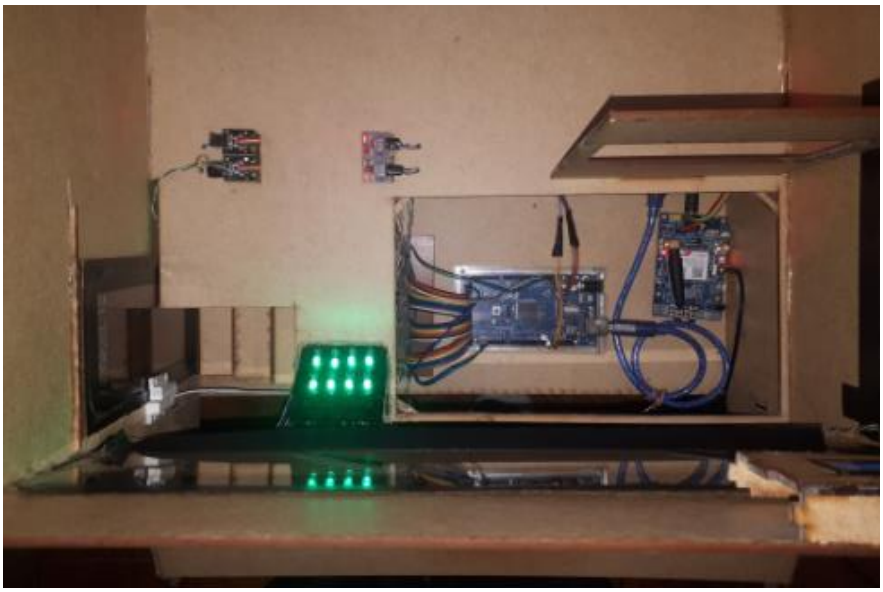


Figura 4.53 Componentes instalados y operativos dentro de la maqueta

Fuente: Elaborado por el autor

4.2.9 Ensamblaje completo.

En la Figura 4.54, se verifica que todo se encuentre operativo y presentable incluso se valida que el sistema está operativo.

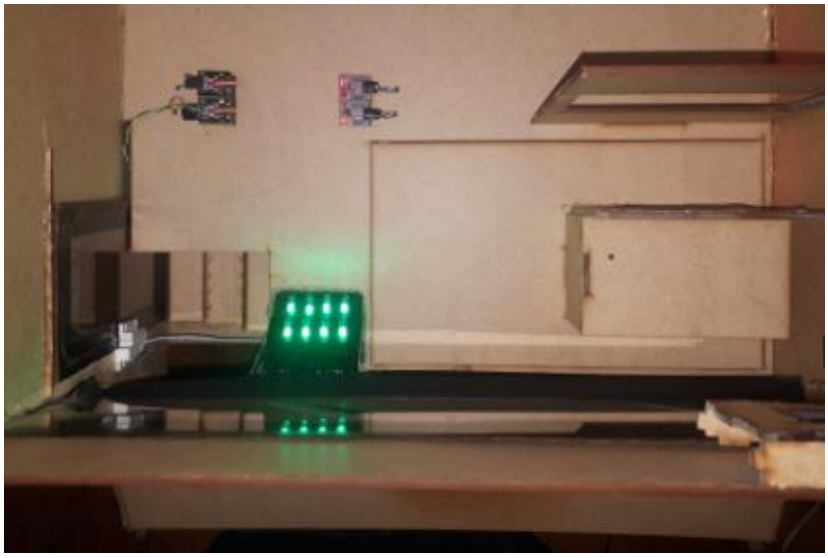


Figura 4.54 Ensamblaje completo del prototipo

Fuente: Elaborado por el autor

Figura 4.55 Ensamblaje completo del prototipo vista frontal

Fuente: Elaborado por el autor

4.3 Pruebas de funcionamiento.

En primera instancia se debe realizar pruebas de encendido y duración de energía de respaldo cuya duración es similar al análisis realizado anteriormente, en el primer encendido el prototipo demora 2 minutos en encenderse completamente ya que mientras no reciba confirmación del módulo SIM808 no empezara a operar.

Inmediatamente se desconecta completamente la alimentación del prototipo y poco después se energiza, en ese momento se evidencia que demora 15 segundos en encenderse y estar operativo completamente y así estar listo para recibir cualquier dato generado por los diferentes dispositivos electrónicos implementados dentro de la maqueta del prototipo

Una vez que se haya comprobado que todo se encuentre operativo se procede a realizar las diferentes pruebas que se presentan a continuación:

.

Se debe valida que todos los interruptores finales de carrera de las butacas se encuentren operativas, se envié correctamente el evento registrado a la base de datos y a su vez le notifiqué al usuario del acontecimiento sucedió.

Se toma un ejemplo de la butaca número 7 y como se puede observar en la Figura 4.56, al momento de abrocharse el cinturón de seguridad inmediatamente se registra el evento en la base de datos en la página de butacas y posteriormente se recibe el SMS al propietario o socio de la unidad.

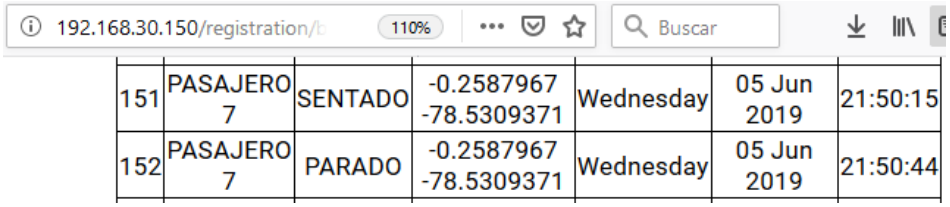


Figura 4.56 Butaca 7, dato recibido en la base de datos

Fuente: Elaborado por el autor.

De la misma manera se evidencia que llega el dato hacia el teléfono celular del socio o dueño de la unidad, incluso llega con las coordenadas del lugar en donde se abrochó el cinturón de seguridad el pasajero con el número de la butaca correspondiente tal como se desarrolló en la programación en la siguiente figura se puede observar dicho resultado.

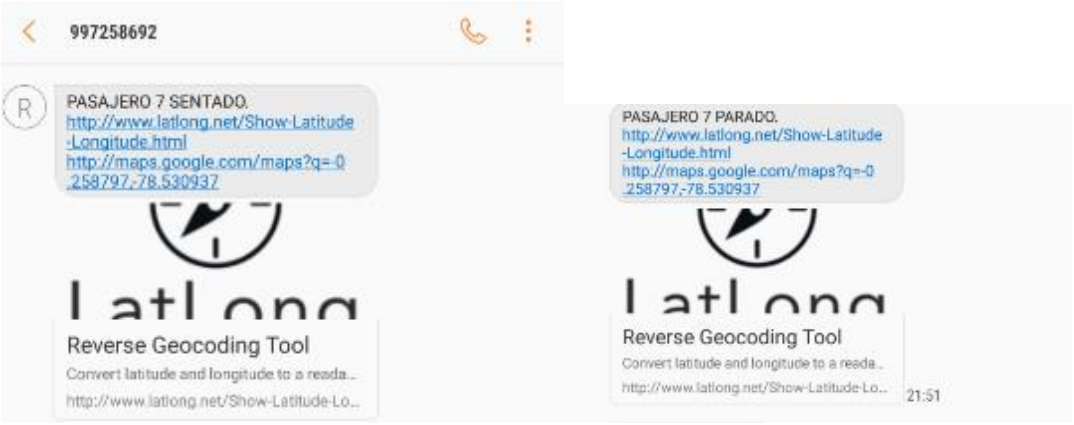


Figura 4.57 Dato SMS recibido de altera en la butaca 7

Fuente: Elaborado por el autor.

Se valida cuando el interruptor magnético entra a funcionar envía el dato de que si la puerta se abre o se cierra incluso si la puerta fue abierta cunado la unidad un se encontraba en movimiento, dicha información llega a la base de datos para poder ser visualizada en tiempo real siempre y cuando la unidad cuente con una conexión hacia internet. En la siguiente figura se evidencia la prueba realizada.

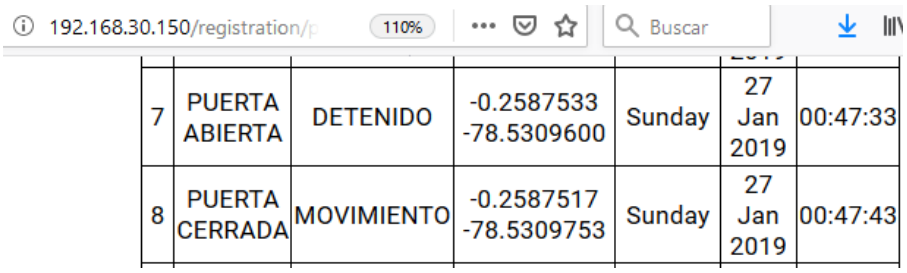


Figura 4.58 Estado de la puerta recibido en la base de datos

Fuente: Elaborado por el autor.

De la misma manera se verifica que cuando pasa a través de los módulos laser este detecta el movimiento que está realizando el pasajero ya sé que suba o baje de la unidad de forma automática y al mismo tiempo registrando dicho evento en la base de datos y enviado un SMS de alerta al usuario o dueño de la unidad si la situación lo a merite, se presenta el resultado obtenido cunado un pasajero aborda la unidad y pasa a través de los módulos laser en la Figura 4.59, mientras que en la Figura 4.60 se evidencia que también llega la alerta SMS al teléfono cedular del dueño o socio de la unidad.

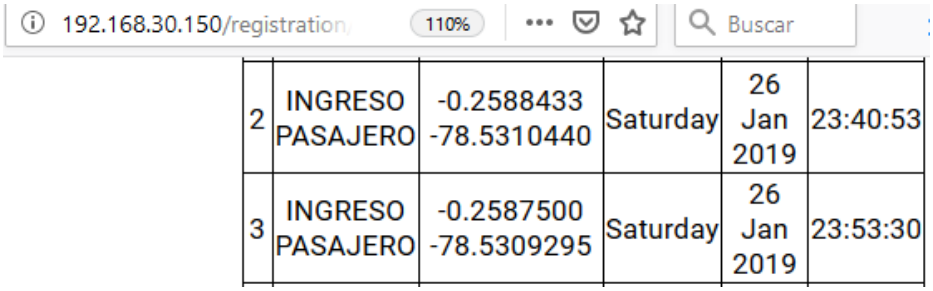


Figura 4.59 Estatus de ingreso de pasajero recibido en la base de datos

Fuente: Elaborado por el autor.

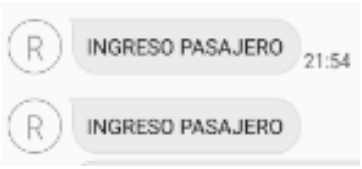


Figura 4.60 Alerta SMS recibida

Fuente: Elaborado por el autor.

4.4 Análisis y resultados.

Después de concluir con la elaboración del prototipo y haber verificado el funcionamiento del mismo, se procede a validar si existe algún inconveniente presentado, con la finalidad de corregirlo y encontrar una pronta solución.

Durante las pruebas realizadas se evidenció que el tiempo de respuesta al momento que se detecta que un pasajero está haciendo uso del cinturón de seguridad hacia la base de datos su registro es inmediato. Por otro lado, el tiempo de respuesta con la que se da a conocer al dueño o socio de la unidad a través de un mensaje corto SMS es de 5 segundos aproximadamente.

Para verificar que el microcomputador esté funcionando correctamente basta con realizar un ping al servidor, si se tiene respuesta del mismo quiere decir que se encuentra operativo, en caso contrario no va a almacenar ningún dato enviado por Arduino Mega 2560. Con los datos recolectados a través del servidor de datos se puede realizar un análisis del

número de paradas realizadas, el número de pasajeros que abordaron y salieron de la unidad y el recorrido que tuvo durante todo el viaje, dicha información siempre estará disponible en cualquier momento.

El primer inicio del prototipo tiene un tiempo considerable de espera, ya que al estar apagado mucho tiempo el módulo SIM808 GSM/GPS tiene receptar la señal GPS del sitio o lugar donde se encuentre en ese momento la unidad, este parámetro no se lo puede modificar según la hoja de datos del fabricante, en el segundo inicio ya se iniciará inmediatamente y se encontrara listo para su funcionamiento.

CONCLUSIONES Y RECOMENDACIONES

Conclusiones

Después del análisis del diseño y construcción del prototipo para el control y monitoreo automático del ingreso y salida de pasajeros en un bus interprovincial con alertas SMS, se presentan las siguientes conclusiones, las que permitirán visualizar de mejor manera su funcionalidad.

- Se desarrolló con éxito el prototipo para el control y monitoreo automático del ingreso y salida de pasajeros en un bus interprovincial con alertas SMS, en este proyecto se relacionó tanto en la parte física como la parte lógica para buscar la correcta funcionalidad del prototipo, brindando de esta manera un estatus en tiempo real de la unidad de transporte. De acuerdo con las pruebas realizadas se tiene una visión más clara del número de personas que suben o bajan de la unidad y lo más importante es la de conocer que cantidad de dinero se recaudó aproximadamente durante el viaje para evitar así reportes económicos falsos.

- Con ayuda de los interruptores finales de carrera se impulsará el uso del cinturón de seguridad dentro de las unidades de transporte, evitando reducir el número personas fallecidas durante un siniestro o accidente por no hacer uso del cinturón de seguridad, además de evitar una multa al conductor y una sanción económica del 15% del salario básico.

- Fue factible acoplar todos los elementos electrónicos y de comunicación entre los diferentes dispositivos que conforman el prototipo para que de esta manera sea autónomo e independiente, mejorando así la administración de la unidad y evitando que el operador reporte cantidades erróneas o falsas.

- Durante las pruebas realizadas se evidenció que la antena GPS debe encontrarse sobre una altura de 180 centímetros sobre el nivel del suelo, garantizando así una mejor precisión al momento de receptar las coordenadas.

- La construcción de la base de datos se la llevo acaba de acuerdo a los requerimientos planteados para la elaboración del prototipo en el cual contendrá información relevante de los diferentes eventos que ocurran dentro de la unidad de transporte.

- La programación realizada para el microcontrolador Arduino Mega 2560, cumplió con las expectativas deseadas para obtener un óptimo funcionamiento de los elementos electrónicos que forman parte del prototipo. De la misma manera para la comunicación con Raspberry PI se logró enviar los datos a través del puerto serial que tiene cada uno.

- Raspberry PI es una gran opción al momento de implementar una base de datos, ya que es un microcomputador de bajo costo y un rendimiento medio para aplicaciones que no requieran tanto procesamiento o memoria, para este prototipo se instalaron las siguientes aplicaciones: Apache, PHP, MySQL, PhpMyAdmin, permitiendo así obtener una administración más sencilla y amigable de la base de datos.

Recomendaciones

- Es de buena práctica sacar un respaldo de la información almacenada en la base de datos de esta manera se tendrá una información más confiable en caso de pérdida o eliminación, ayudando así a poder reconstruirla en el momento que lo requiera.

- Para prevenir fallos de operación en el microcomputador se recomienda una alimentación de 5V a 2A fijos con una tarjeta de memoria de 8 GB de capacidad recomendada por el fabricante, asegurando así que el prototipo funcione a su 100% de capacidad.

- En un futuro se podría desarrollar una opción de descargar de la información en la página web implementada, de tal manera que se podría generar credenciales de acceso a la persona que lo necesite y puedan utilizar dicha información en forma que a bien tuviere.

- Se recomienda una revisión periódica de las baterías que alimentan al prototipo en caso de que se llegara a cortar la energía principal, también verificar el estado de los interruptores finales de carrera ya que podrían sufrir un desgaste por el uso continuo del cinturón de seguridad, de esta manera prevenir un mal funcionamiento del prototipo.

REFERENCIAS BIBLIOGRÁFICAS

Agencia Nacional de Transito. (14 de Noviembre de 2016). *Agencia Nacional de Transito*. Obtenido de REGLAMENTO A LEY DE TRANSPORTE TERRESTRE: https://www.ant.gob.ec/index.php/ant/base-legal/reglamento-general-para-la-aplicacion-de-la-lotttsv

Ardumotive. (12 de Junio de 2018). *How to use an I2CLCD display*. Obtenido de http://www.ardumotive.com

Boylestad, R. (2009). *Electrónica: teoria de circuitos y dispositivos electrónicos.* México: PEARSON.

Busae. (22 de Junio de 2018). *Software de Transporte de Pasajeros*. Obtenido de https://www.busae.com/

Dualtronica. (21 de Junio de 2018). *Diodo laser 650nm 5mW*. Obtenido de https://dualtronica.com

Ecured. (18 de Junio de 2018). *PhpMyAdmin*. Obtenido de https://www.ecured.cu/

El Comercio. (18 de Diciembre de 2018). *ANT prohíbe circular a buses sin cinturones de seguridad en el Ecuador*. Obtenido de https://www.elcomercio.com

FENACOTIP. (23 de Mayo de 2018). *Unidades de Transporte*. Obtenido de http://fenacotip.com.ec

Geekelectronica. (12 de Junio de 2018). *¿QUÉ ES PYTHON?* Obtenido de https://geekelectronica.com

Grupo El Comercio. (4 de Octubre de 2018). Nueve de cada 10 admiten que no se ponen el cinturón en transporte público. *El Comercio*, pág. 2.

Hostinger. (22 de Junio de 2018). *¿Qué es Apache?* Obtenido de https://www.hostinger.es

ItSoftware. (22 de Junio de 2018). *¿Qué es y para que sirve MySQL Database?* Obtenido de https://itsoftware.com.co

Maker Pro. (17 de Octubre de 2018). *Build a Car Tracking System with the SIM808 Module*. Obtenido de Build a Car Tracking System with the SIM808 Module: https://maker.pro/

PHP. (22 de Junio de 2018). *¿Qué es PHP?* Obtenido de https://www.php.net

Prometec. (Julio de 2018). *SIM808: GSM/GPRS + GPS*. Obtenido de www.prometec.net

Quiminet. (22 de Junio de 2018). *¿Qué son los interruptores finales de carrera?* Obtenido de https://www.quiminet.com

Raspberry PI. (22 de Junio de 2018). *Acerca de Raspberry PI 3*. Obtenido de https://www.raspberrypi.org

Raspbian France . (20 de Noviembre de 2018). *How to install a web server on the Raspberry Pi*. Obtenido de How to install a web server on the Raspberry Pi: https://howtoraspberrypi.com/

Santos González, M. (2014). *Sistemas Telematicos.* Madrid, España: RA-MA.

Savant, C., Roden, M., & Carperter, G. (2000). *Diseño Electrónico, Circuitos y Sistemas.* Mexico: Pearson Education.

Siasa. (22 de Junio de 2018). *Torniquetes*. Obtenido de https://www.siasa.com

Sol Llaven, D. (2015). *Sistemas operativos: panorama para la ingeniería en computación e informática.* Colina San Juan Tlihuaca Azcapotzalco: Grupo Editorial Patria.

Tecmikro. (20 de Octubre de 2018). *Arduino Mega 2560*. Obtenido de Arduino Mega 2560: https://tecmikro.com/tarjetas-programables/332-mega-2560.html

Ubunlog. (12 de Junio de 2018). *MySQL Workbench, herramienta visual para el diseño de bases de datos*. Obtenido de https://ubunlog.com

Villada Romero, J. L. (2015). *Instalación y configuración del software de servidor web UF1271* (Primera ed.). Antequera: IC Editorial.

Vistronica. (22 de Junio de 2018). *Módulo Receptor Laser*. Obtenido de https://www.vistronica.com

ANEXOS

ANEXO 1. CÓDIGO DE PROGRAMACIÓN ARDUINO

```
//PIN 10 DEL ARDUINO MEGA AL TXD DE LA PRIMERA FILA DEL SIM808
//PIN 11 DEL ARDUINO MEGA AL RXD DE LA PRIMERA FILA DEL SIM808
//ALIMENTAR AL SIM808 CON 3.3V Y GND SACADOS DEL ARDUINO MEGA

#include <DFRobot_sim808.h>
#include <SoftwareSerial.h>

#include <Wire.h>
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C LCD(0X3F,16,2);

#define PIN_TX    10
#define PIN_RX    11

SoftwareSerial mySerial(PIN_TX,PIN_RX);
DFRobot_SIM808 sim808(&mySerial);

char MESSAGE[300];
char lat[12];
char lon[12];

#define PHONE_NUMBER "0990257279"
#define MENSAJE_PUERTA "LA PUERTA SE ABRIO"
#define MENSAJE_PUERTC "LA PUERTA SE CERRO"
#define MENSAJE_PASAJERO "INGRESO PASAJERO"
#define MENSAJE_PASAJERO_S "SALIENDO PASAJERO"

#define MESSAGE_LENGTH 160
char message[MESSAGE_LENGTH];
int messageIndex = 0;
char phone[16];
char datetime[24];

int FCA1=22;
int FCA2=23;
int FCA3=24;
int FCA4=25;
int FCA5=26;
int FCA6=27;
int FCA7=28;
int FCA8=29;

int val_FCA1;
int val_FCA2;
int val_FCA3;
int val_FCA4;
int val_FCA5;
int val_FCA6;
int val_FCA7;
int val_FCA8;

int led_verde_A1=30;
int led_verde_A2=31;
int led_verde_A3=32;
int led_verde_A4=33;
int led_verde_A5=34;
int led_verde_A6=35;
int led_verde_A7=36;
int led_verde_A8=37;

int led_rojo_A1=38;
int led_rojo_A2=39;
int led_rojo_A3=40;
int led_rojo_A4=41;
int led_rojo_A5=42;
int led_rojo_A6=43;
int led_rojo_A7=44;
int led_rojo_A8=45;

byte sensor_mag=46;
```

```
byte val_sensor_mag = 0;
byte estadoBotonAnt = 0;

int sensor_e1=47;
int sensor_e2=48;
int val_sensor_e1;
int val_sensor_e2;

int pA1=49;
int val_pA1;

int cont1=0;
int cont2=0;

int cont3=0;
int cont4=0;

int cont5=0;
int cont6=0;

int cont7=0;
int cont8=0;

int cont9=0;
int cont10=0;

int cont11=0;
int cont12=0;

int cont13=0;
int cont14=0;

int cont15=0;
int cont16=0;

int aux=0;

void setup()
{
  mySerial.begin(9600);
  Serial.begin(9600);
  LCD.backlight();
  LCD.init();

  pinMode(FCA1,INPUT_PULLUP);
  pinMode(FCA2,INPUT_PULLUP);
  pinMode(FCA3,INPUT_PULLUP);
  pinMode(FCA4,INPUT_PULLUP);
  pinMode(FCA5,INPUT_PULLUP);
  pinMode(FCA6,INPUT_PULLUP);
  pinMode(FCA7,INPUT_PULLUP);
  pinMode(FCA8,INPUT_PULLUP);

  pinMode(led_verde_A1,OUTPUT);
  pinMode(led_verde_A2,OUTPUT);
  pinMode(led_verde_A3,OUTPUT);
  pinMode(led_verde_A4,OUTPUT);
  pinMode(led_verde_A5,OUTPUT);
  pinMode(led_verde_A6,OUTPUT);
  pinMode(led_verde_A7,OUTPUT);
  pinMode(led_verde_A8,OUTPUT);

  digitalWrite(led_verde_A1,LOW);
  digitalWrite(led_verde_A2,LOW);
  digitalWrite(led_verde_A3,LOW);
  digitalWrite(led_verde_A4,LOW);
  digitalWrite(led_verde_A5,LOW);
  digitalWrite(led_verde_A6,LOW);
  digitalWrite(led_verde_A7,LOW);
  digitalWrite(led_verde_A8,LOW);
```

```
  pinMode(led_rojo_A1,OUTPUT);
  pinMode(led_rojo_A2,OUTPUT);
  pinMode(led_rojo_A3,OUTPUT);
  pinMode(led_rojo_A4,OUTPUT);
  pinMode(led_rojo_A5,OUTPUT);
  pinMode(led_rojo_A6,OUTPUT);
  pinMode(led_rojo_A7,OUTPUT);
  pinMode(led_rojo_A8,OUTPUT);

  digitalWrite(led_rojo_A1,LOW);
  digitalWrite(led_rojo_A2,LOW);
  digitalWrite(led_rojo_A3,LOW);
  digitalWrite(led_rojo_A4,LOW);
  digitalWrite(led_rojo_A5,LOW);
  digitalWrite(led_rojo_A6,LOW);
  digitalWrite(led_rojo_A7,LOW);
  digitalWrite(led_rojo_A8,LOW);

  pinMode(sensor_mag,INPUT_PULLUP);

  while(!sim808.init()) {
    delay(1000);
    LCD.clear();
    LCD.setCursor(0,0);
    LCD.print("SIM808:");
    LCD.setCursor(0,1);
    LCD.print("ERROR CONEXION");
    Serial.println("ERROR CONEXION");
  }

  if( sim808.attachGPS()){
    LCD.clear();
    LCD.setCursor(0,0);
    LCD.print("GSM:INICIADO");
    LCD.setCursor(0,1);
    LCD.print("GPS:INICIADO");
    delay(2000);
    }
  else {
    LCD.clear();
    LCD.setCursor(0,0);
    LCD.print("GPS:ERROR");
}
}


void loop()
{
  if (sim808.getGPS()) {

    leer_FC();
    enviar_RB();
    //ENVÍA UN 0 CUANDO ESTA UNIDA, Y 1 CUANDO ESTA SEPARADO
    val_sensor_mag=digitalRead(sensor_mag);
    if(val_sensor_mag==HIGH && estadoBotonAnt == LOW){
      enviar_mensaje1();
    }
    if(val_sensor_mag==LOW && estadoBotonAnt == HIGH){
      enviar_mensaje3();
    }
    estadoBotonAnt = val_sensor_mag;

    //ENVÍA UN 0 CUANDO ESTA SIN EL LÁSER Y 1 CUANDO EL LÁSER LE APUNTA.
    val_sensor_e1=digitalRead(sensor_e1);
    val_sensor_e2=digitalRead(sensor_e2);
    if(val_sensor_e1==LOW){
      if(val_sensor_e2==LOW){
        enviar_mensaje2();
      }
```

```
    }
    }
}

void leer_FC(){
//ENVÍA UN 1 CUANDO ESTA SIN PRESIONAR, Y 0 CUANDO ESTA PRESIONADO
  val_FCA1=digitalRead(FCA1);
  val_FCA2=digitalRead(FCA2);
  val_FCA3=digitalRead(FCA3);
  val_FCA4=digitalRead(FCA4);
  val_FCA5=digitalRead(FCA5);
  val_FCA6=digitalRead(FCA6);
  val_FCA7=digitalRead(FCA7);
  val_FCA8=digitalRead(FCA8);

int v[] = {val_FCA1, val_FCA2, val_FCA3, val_FCA4, val_FCA5, val_FCA6, val_FCA7, val_FCA8};
int n=0;
int i=0;
int libres;

for(i=0;i<=7;i++){
  if(v[i]==0){
    n++;
  }
}

libres=8-n;

  LCD.clear();
  LCD.setCursor(4,0);
  LCD.print("ASIENTOS");
  LCD.setCursor(0,1);
  LCD.print("LIB:");
  LCD.setCursor(4,1);
  LCD.print(libres);
  LCD.setCursor(8,1);
  LCD.print("OCU:");
  LCD.setCursor(13,1);
  LCD.print(n);
  delay(100);

  if(val_FCA1==LOW){
    digitalWrite(led_rojo_A1,HIGH);
    digitalWrite(led_verde_A1,LOW);
    if (cont1==0 && aux==1){
    cont2=0;
    Serial.print("PASAJERO 1         ");
    enviar_mensaje_p1sentado();
    }
    cont1++;
  }
  else{
    digitalWrite(led_rojo_A1,LOW);
    digitalWrite(led_verde_A1,HIGH);
    if (cont2==0 && aux==1){
    cont1=0;
    Serial.print("PASAJERO 1         ");
    enviar_mensaje_p1parado();
    }
    cont2++;
  }

  if(val_FCA2==LOW){
    digitalWrite(led_rojo_A2,HIGH);
    digitalWrite(led_verde_A2,LOW);
    if (cont3==0 && aux==1){
    cont4=0;
    Serial.print("PASAJERO 2         ");
    enviar_mensaje_p2sentado();
    }
```

```
  cont3++;
}
else{
  digitalWrite(led_rojo_A2,LOW);
  digitalWrite(led_verde_A2,HIGH);
  if (cont4==0 && aux==1){
  cont3=0;
  Serial.print("PASAJERO 2        ");
  enviar_mensaje_p2parado();
  }
  cont4++;
}

if(val_FCA3==LOW){
  digitalWrite(led_rojo_A3,HIGH);
  digitalWrite(led_verde_A3,LOW);
  if (cont5==0 && aux==1){
  cont6=0;
  Serial.print("PASAJERO 3        ");
  enviar_mensaje_p3sentado();
  }
  cont5++;
}
else{
  digitalWrite(led_rojo_A3,LOW);
  digitalWrite(led_verde_A3,HIGH);
  if (cont6==0 && aux==1){
  cont5=0;
  Serial.print("PASAJERO 3        ");
  enviar_mensaje_p3parado();
  }
  cont6++;
}

if(val_FCA4==LOW){
  digitalWrite(led_rojo_A4,HIGH);
  digitalWrite(led_verde_A4,LOW);
  if (cont7==0 && aux==1){
  cont8=0;
  Serial.print("PASAJERO 4        ");
  enviar_mensaje_p4sentado();
  }
  cont7++;
}
else{
  digitalWrite(led_rojo_A4,LOW);
  digitalWrite(led_verde_A4,HIGH);
  if (cont8==0 && aux==1){
  cont7=0;
  Serial.print("PASAJERO 4        ");
  enviar_mensaje_p4parado();
  }
  cont8++;
}

if(val_FCA5==LOW){
  digitalWrite(led_rojo_A5,HIGH);
  digitalWrite(led_verde_A5,LOW);
  if (cont9==0 && aux==1){
  cont10=0;
  Serial.print("PASAJERO 5        ");
  enviar_mensaje_p5sentado();
  }
  cont9++;
}
else{
  digitalWrite(led_rojo_A5,LOW);
  digitalWrite(led_verde_A5,HIGH);
  if (cont10==0 && aux==1){
  cont9=0;
```

```
  Serial.print("PASAJERO 5        ");
  enviar_mensaje_p5parado();
  }
  cont10++;
 }

 if(val_FCA6==LOW){
  digitalWrite(led_rojo_A6,HIGH);
  digitalWrite(led_verde_A6,LOW);
  if (cont11==0 && aux==1){
  cont12=0;
  Serial.print("PASAJERO 6        ");
  enviar_mensaje_p6sentado();
  }
  cont11++;
 }
 else{
  digitalWrite(led_rojo_A6,LOW);
  digitalWrite(led_verde_A6,HIGH);
  if (cont12==0 && aux==1){
  cont11=0;
  Serial.print("PASAJERO 6        ");
  enviar_mensaje_p6parado();
  }
  cont12++;
 }

 if(val_FCA7==LOW){
  digitalWrite(led_rojo_A7,HIGH);
  digitalWrite(led_verde_A7,LOW);
  if (cont13==0 && aux==1){
  cont14=0;
  Serial.print("PASAJERO 7        ");
  enviar_mensaje_p7sentado();
  }
  cont13++;
 }
 else{
  digitalWrite(led_rojo_A7,LOW);
  digitalWrite(led_verde_A7,HIGH);
  if (cont14==0 && aux==1){
  cont13=0;
  Serial.print("PASAJERO 7        ");
  enviar_mensaje_p7parado();
  }
  cont14++;
 }

 if(val_FCA8==LOW){
  digitalWrite(led_rojo_A8,HIGH);
  digitalWrite(led_verde_A8,LOW);
  if (cont15==0 && aux==1){
  cont16=0;
  Serial.print("PASAJERO 8        ");
  enviar_mensaje_p8sentado();
  }
  cont15++;
 }
 else{
  digitalWrite(led_rojo_A8,LOW);
  digitalWrite(led_verde_A8,HIGH);
  if (cont16==0 && aux==1){
  cont15=0;
  Serial.print("PASAJERO 8        ");
  enviar_mensaje_p8parado();
  }
  cont16++;
 }
 aux=1;
}
```

```
void enviar_mensaje1(){
     Serial.print("PUERTA ABIERTA     ");

     float la = sim808.GPSdata.lat;
     float lo = sim808.GPSdata.lon;

     dtostrf(la, 8, 6, lat);
     dtostrf(lo, 8, 6, lon);

     Serial.print(sim808.GPSdata.lat, 7);
     Serial.print(" ");
     Serial.print(sim808.GPSdata.lon, 7);
     Serial.print(" ");
     //Serial.print(sim808.GPSdata.speed_kph);
     if (sim808.GPSdata.speed_kph < 0.3){
       Serial.println("DETENIDO");
       }
     else {
       Serial.println("MOVIMIENTO");
       }

     delay(1000);
     sim808.sendSMS(PHONE_NUMBER,MENSAJE_PUERTA);
     delay(1000);
     //Serial.println("MENSAJE ENVIADO LA PUERTA SE ABRIO");
}

void enviar_mensaje2(){
     Serial.print("INGRESO PASAJERO   ");

     float la = sim808.GPSdata.lat;
     float lo = sim808.GPSdata.lon;

     dtostrf(la, 8, 6, lat);
     dtostrf(lo, 8, 6, lon);

     Serial.print(sim808.GPSdata.lat, 7);
     Serial.print(" ");
     Serial.println(sim808.GPSdata.lon, 7);

     delay(1000);
     sim808.sendSMS(PHONE_NUMBER,MENSAJE_PASAJERO);
     delay(1000);
     //Serial.println("MENSAJE ENVIADO INGRESO PASAJERO");
}

void enviar_mensaje3(){
     Serial.print("PUERTA CERRADA     ");

     float la = sim808.GPSdata.lat;
     float lo = sim808.GPSdata.lon;

     dtostrf(la, 8, 6, lat);
     dtostrf(lo, 8, 6, lon);

     Serial.print(sim808.GPSdata.lat, 7);
     Serial.print(" ");
     Serial.print(sim808.GPSdata.lon, 7);
     Serial.print(" ");
     //Serial.print(sim808.GPSdata.speed_kph);
     if (sim808.GPSdata.speed_kph < 0.3){
       Serial.println("DETENIDO");
       }
     else {
       Serial.println("MOVIMIENTO");
       }

     delay(1000);
     sim808.sendSMS(PHONE_NUMBER,MENSAJE_PUERTC);
```

```
      delay(1000);
      //Serial.println("MENSAJE ENVIADO LA PUERTA SE CERRO");
}


void enviar_mensaje_p1sentado(){

    float la = sim808.GPSdata.lat;
    float lo = sim808.GPSdata.lon;

    dtostrf(la, 8, 6, lat);
    dtostrf(lo, 8, 6, lon);

    Serial.print(sim808.GPSdata.lat, 7);
    Serial.print(" ");
    Serial.print(sim808.GPSdata.lon, 7);
    Serial.print(" ");
    Serial.println("SENTADO ");

    sprintf(MESSAGE, "PASAJERO 1 SENTADO.\nhttp://www.latlong.net/Show-Latitude-
Longitude.html\nhttp://maps.google.com/maps?q=%s,%s\n",lat,lon);

    delay(1000);
    sim808.sendSMS(PHONE_NUMBER,MESSAGE);
    delay(1000);
}

void enviar_mensaje_p1parado(){

    float la = sim808.GPSdata.lat;
    float lo = sim808.GPSdata.lon;

    dtostrf(la, 8, 6, lat);
    dtostrf(lo, 8, 6, lon);

    Serial.print(sim808.GPSdata.lat, 7);
    Serial.print(" ");
    Serial.print(sim808.GPSdata.lon, 7);
    Serial.print(" ");
    Serial.println("PARADO  ");

    sprintf(MESSAGE, "PASAJERO 1 PARADO.\nhttp://www.latlong.net/Show-Latitude-
Longitude.html\nhttp://maps.google.com/maps?q=%s,%s\n",lat,lon);

    delay(1000);
    sim808.sendSMS(PHONE_NUMBER,MESSAGE);
    delay(1000);

}

void enviar_mensaje_p2sentado(){

    float la = sim808.GPSdata.lat;
    float lo = sim808.GPSdata.lon;

    dtostrf(la, 8, 6, lat);
    dtostrf(lo, 8, 6, lon);

    Serial.print(sim808.GPSdata.lat, 7);
    Serial.print(" ");
    Serial.print(sim808.GPSdata.lon, 7);
    Serial.print(" ");
    Serial.println("SENTADO ");

    sprintf(MESSAGE, "PASAJERO 2 SENTADO.\nhttp://www.latlong.net/Show-Latitude-
Longitude.html\nhttp://maps.google.com/maps?q=%s,%s\n",lat,lon);

    delay(1000);
    sim808.sendSMS(PHONE_NUMBER,MESSAGE);
    delay(1000);
```

```
}

void enviar_mensaje_p2parado(){

    float la = sim808.GPSdata.lat;
    float lo = sim808.GPSdata.lon;

    dtostrf(la, 8, 6, lat);
    dtostrf(lo, 8, 6, lon);

    Serial.print(sim808.GPSdata.lat, 7);
    Serial.print(" ");
    Serial.print(sim808.GPSdata.lon, 7);
    Serial.print(" ");
    Serial.println("PARADO  ");

    sprintf(MESSAGE, "PASAJERO 2 PARADO.\nhttp://www.latlong.net/Show-Latitude-
Longitude.html\nhttp://maps.google.com/maps?q=%s,%s\n",lat,lon);

    delay(1000);
    sim808.sendSMS(PHONE_NUMBER,MESSAGE);
    delay(1000);

}

void enviar_mensaje_p3sentado(){

    float la = sim808.GPSdata.lat;
    float lo = sim808.GPSdata.lon;

    dtostrf(la, 8, 6, lat);
    dtostrf(lo, 8, 6, lon);

    Serial.print(sim808.GPSdata.lat, 7);
    Serial.print(" ");
    Serial.print(sim808.GPSdata.lon, 7);
    Serial.print(" ");
    Serial.println("SENTADO ");

    sprintf(MESSAGE, "PASAJERO 3 SENTADO.\nhttp://www.latlong.net/Show-Latitude-
Longitude.html\nhttp://maps.google.com/maps?q=%s,%s\n",lat,lon);

    delay(1000);
    sim808.sendSMS(PHONE_NUMBER,MESSAGE);
    delay(1000);

}

void enviar_mensaje_p3parado(){

    float la = sim808.GPSdata.lat;
    float lo = sim808.GPSdata.lon;

    dtostrf(la, 8, 6, lat);
    dtostrf(lo, 8, 6, lon);

    Serial.print(sim808.GPSdata.lat, 7);
    Serial.print(" ");
    Serial.print(sim808.GPSdata.lon, 7);
    Serial.print(" ");
    Serial.println("PARADO  ");

    sprintf(MESSAGE, "PASAJERO 3 PARADO.\nhttp://www.latlong.net/Show-Latitude-
Longitude.html\nhttp://maps.google.com/maps?q=%s,%s\n",lat,lon);

    delay(1000);
    sim808.sendSMS(PHONE_NUMBER,MESSAGE);
    delay(1000);
```

```
}

void enviar_mensaje_p4sentado(){

    float la = sim808.GPSdata.lat;
    float lo = sim808.GPSdata.lon;

    dtostrf(la, 8, 6, lat);
    dtostrf(lo, 8, 6, lon);

    Serial.print(sim808.GPSdata.lat, 7);
    Serial.print(" ");
    Serial.print(sim808.GPSdata.lon, 7);
    Serial.print(" ");
    Serial.println("SENTADO ");

    sprintf(MESSAGE, "PASAJERO 4 SENTADO.\nhttp://www.latlong.net/Show-Latitude-
Longitude.html\nhttp://maps.google.com/maps?q=%s,%s\n",lat,lon);

    delay(1000);
    sim808.sendSMS(PHONE_NUMBER,MESSAGE);
    delay(1000);

}

void enviar_mensaje_p4parado(){

    float la = sim808.GPSdata.lat;
    float lo = sim808.GPSdata.lon;

    dtostrf(la, 8, 6, lat);
    dtostrf(lo, 8, 6, lon);

    Serial.print(sim808.GPSdata.lat, 7);
    Serial.print(" ");
    Serial.print(sim808.GPSdata.lon, 7);
    Serial.print(" ");
    Serial.println("PARADO  ");

    sprintf(MESSAGE, "PASAJERO 4 PARADO.\nhttp://www.latlong.net/Show-Latitude-
Longitude.html\nhttp://maps.google.com/maps?q=%s,%s\n",lat,lon);

    delay(1000);
    sim808.sendSMS(PHONE_NUMBER,MESSAGE);
    delay(1000);

}

void enviar_mensaje_p5sentado(){

    float la = sim808.GPSdata.lat;
    float lo = sim808.GPSdata.lon;

    dtostrf(la, 8, 6, lat);
    dtostrf(lo, 8, 6, lon);

    Serial.print(sim808.GPSdata.lat, 7);
    Serial.print(" ");
    Serial.print(sim808.GPSdata.lon, 7);
    Serial.print(" ");
    Serial.println("SENTADO ");

    sprintf(MESSAGE, "PASAJERO 5 SENTADO.\nhttp://www.latlong.net/Show-Latitude-
Longitude.html\nhttp://maps.google.com/maps?q=%s,%s\n",lat,lon);

    delay(1000);
    sim808.sendSMS(PHONE_NUMBER,MESSAGE);
    delay(1000);

}
```

```
void enviar_mensaje_p5parado(){

    float la = sim808.GPSdata.lat;
    float lo = sim808.GPSdata.lon;

    dtostrf(la, 8, 6, lat);
    dtostrf(lo, 8, 6, lon);

    Serial.print(sim808.GPSdata.lat, 7);
    Serial.print(" ");
    Serial.print(sim808.GPSdata.lon, 7);
    Serial.print(" ");
    Serial.println("PARADO  ");

    sprintf(MESSAGE, "PASAJERO 5 PARADO.\nhttp://www.latlong.net/Show-Latitude-
Longitude.html\nhttp://maps.google.com/maps?q=%s,%s\n",lat,lon);

    delay(1000);
    sim808.sendSMS(PHONE_NUMBER,MESSAGE);
    delay(1000);

}

void enviar_mensaje_p6sentado(){

    float la = sim808.GPSdata.lat;
    float lo = sim808.GPSdata.lon;

    dtostrf(la, 8, 6, lat);
    dtostrf(lo, 8, 6, lon);

    Serial.print(sim808.GPSdata.lat, 7);
    Serial.print(" ");
    Serial.print(sim808.GPSdata.lon, 7);
    Serial.print(" ");
    Serial.println("SENTADO ");

    sprintf(MESSAGE, "PASAJERO 6 SENTADO.\nhttp://www.latlong.net/Show-Latitude-
Longitude.html\nhttp://maps.google.com/maps?q=%s,%s\n",lat,lon);

    delay(1000);
    sim808.sendSMS(PHONE_NUMBER,MESSAGE);
    delay(1000);

}

void enviar_mensaje_p6parado(){

    float la = sim808.GPSdata.lat;
    float lo = sim808.GPSdata.lon;

    dtostrf(la, 8, 6, lat);
    dtostrf(lo, 8, 6, lon);

    Serial.print(sim808.GPSdata.lat, 7);
    Serial.print(" ");
    Serial.print(sim808.GPSdata.lon, 7);
    Serial.print(" ");
    Serial.println("PARADO  ");

    sprintf(MESSAGE, "PASAJERO 6 PARADO.\nhttp://www.latlong.net/Show-Latitude-
Longitude.html\nhttp://maps.google.com/maps?q=%s,%s\n",lat,lon);

    delay(1000);
    sim808.sendSMS(PHONE_NUMBER,MESSAGE);
    delay(1000);

}
```

```
void enviar_mensaje_p7sentado(){

   float la = sim808.GPSdata.lat;
   float lo = sim808.GPSdata.lon;

   dtostrf(la, 8, 6, lat);
   dtostrf(lo, 8, 6, lon);

   Serial.print(sim808.GPSdata.lat, 7);
   Serial.print(" ");
   Serial.print(sim808.GPSdata.lon, 7);
   Serial.print(" ");
   Serial.println("SENTADO ");

   sprintf(MESSAGE, "PASAJERO 7 SENTADO.\nhttp://www.latlong.net/Show-Latitude-
Longitude.html\nhttp://maps.google.com/maps?q=%s,%s\n",lat,lon);

   delay(1000);
   sim808.sendSMS(PHONE_NUMBER,MESSAGE);
   delay(1000);

}

void enviar_mensaje_p7parado(){

   float la = sim808.GPSdata.lat;
   float lo = sim808.GPSdata.lon;

   dtostrf(la, 8, 6, lat);
   dtostrf(lo, 8, 6, lon);

   Serial.print(sim808.GPSdata.lat, 7);
   Serial.print(" ");
   Serial.print(sim808.GPSdata.lon, 7);
   Serial.print(" ");
   Serial.println("PARADO  ");

   sprintf(MESSAGE, "PASAJERO 7 PARADO.\nhttp://www.latlong.net/Show-Latitude-
Longitude.html\nhttp://maps.google.com/maps?q=%s,%s\n",lat,lon);

   delay(1000);
   sim808.sendSMS(PHONE_NUMBER,MESSAGE);
   delay(1000);

}

void enviar_mensaje_p8sentado(){

   float la = sim808.GPSdata.lat;
   float lo = sim808.GPSdata.lon;

   dtostrf(la, 8, 6, lat);
   dtostrf(lo, 8, 6, lon);

   Serial.print(sim808.GPSdata.lat, 7);
   Serial.print(" ");
   Serial.print(sim808.GPSdata.lon, 7);
   Serial.print(" ");
   Serial.println("SENTADO ");

   sprintf(MESSAGE, "PASAJERO 8 SENTADO.\nhttp://www.latlong.net/Show-Latitude-
Longitude.html\nhttp://maps.google.com/maps?q=%s,%s\n",lat,lon);

   delay(1000);
   sim808.sendSMS(PHONE_NUMBER,MESSAGE);
   delay(1000);

}

void enviar_mensaje_p8parado(){
```

```
    float la = sim808.GPSdata.lat;
    float lo = sim808.GPSdata.lon;

    dtostrf(la, 8, 6, lat);
    dtostrf(lo, 8, 6, lon);

    Serial.print(sim808.GPSdata.lat, 7);
    Serial.print(" ");
    Serial.print(sim808.GPSdata.lon, 7);
    Serial.print(" ");
    Serial.println("PARADO  ");

    sprintf(MESSAGE, "PASAJERO 8 PARADO.\nhttp://www.latlong.net/Show-Latitude-
Longitude.html\nhttp://maps.google.com/maps?q=%s,%s\n",lat,lon);

    delay(1000);
    sim808.sendSMS(PHONE_NUMBER,MESSAGE);
    delay(1000);

}

void enviar_RB(){

  val_FCA1=digitalRead(FCA1);
  val_FCA2=digitalRead(FCA2);
  val_FCA3=digitalRead(FCA3);
  val_FCA4=digitalRead(FCA4);
  val_FCA5=digitalRead(FCA5);
  val_FCA6=digitalRead(FCA6);
  val_FCA7=digitalRead(FCA7);
  val_FCA8=digitalRead(FCA8);
  //ENVÍA UN 0 CUANDO ESTA UNIDA, Y 1 CUANDO ESTA SEPARADO
  val_sensor_mag=digitalRead(sensor_mag);
  //ENVÍA UN 0 CUANDO ESTA SIN EL LÁSER Y 1 CUANDO EL LÁSER LE APUNTA.
  val_sensor_e1=digitalRead(sensor_e1);
  val_sensor_e2=digitalRead(sensor_e2);

  if (Serial.available()){
    char c = Serial.read();
    if (c == 'A'){
      Serial.println("ESTADO DE LOS ASIENTOS");
      Serial.println(val_FCA1);
      Serial.println(val_FCA2);
      Serial.println(val_FCA3);
      Serial.println(val_FCA4);
      Serial.println(val_FCA5);
      Serial.println(val_FCA6);
      Serial.println(val_FCA7);
      Serial.println(val_FCA8);
      }

      else if (c == 'B'){
        Serial.println("ESTADO DE LA PUERTA");
      if(val_sensor_mag==LOW){
        Serial.println("PUERTA CERRADA");
      }
      if(val_sensor_mag==HIGH){
        Serial.println("PUERTA ABIERTA");
      }
      }

      else if (c == 'C'){
        Serial.println("ESTADO SENSOR INGRESO 1");
      if(val_sensor_e1==LOW){
        Serial.println("SENSOR 1 DETECTA PRESENCIA");
      }
      if(val_sensor_e1==HIGH){
        Serial.println("SENSOR 1 NO DETECTA PRESENCIA");
      }
```

```
      Serial.println("ESTADO SENSOR INGRESO 2");
      if(val_sensor_e2==LOW){
        Serial.println("SENSOR 2 DETECTA PRESENCIA");
      }
      if(val_sensor_e2==HIGH){
        Serial.println("SENSOR 2 NO DETECTA PRESENCIA");
      }
      }

      else{
        Serial.println("LETRA NO RECONOCIDA");
        }
        }
}
```

ANEXO 2. CÓDIGO DE PROGRAMACIÓN PYTHON

```
import glob
import time
import commands
import os
import socket
import serial
import mysql.connector

arduino = serial.Serial('/dev/ttyACM0', baudrate=9600)
print("Starting!")

while True:
	line = arduino.readline().strip()
		g = line[0:6]
		a = line[0:18]
		b = line[19:41]
		c = line[42:52]
		d = time.strftime("%A")
		e = time.strftime("%d %b %Y")
		f = time.strftime("%X")

		print(a)
		print(b)
		print(c)
		print(d)
		print(e)
		print(f)

		w = "PASAJE"
		if (g == w):
			dato = {'user':'admin','password':'root$$2018','database':'Transportes','host':'localhost'}
			conexion = mysql.connector.connect(** dato)
			cursor = conexion.cursor()
			valores = ("""INSERT INTO butacas (asiento,estado,coordenadas,dia,fecha,hora) VALUES
(%s,%s,%s,%s,%s,%s)""",(a,c,b,d,e,f))
			try:
				cursor.execute(*valores)
				conexion.commit()
				print("Se ha insertado los valores dentro de la base de datos =)")
			except:
				print("No se ha podido insertar los valores dentro de la base de datos")
			conexion.close()
#-----------------------------------------------------------------
		x = "INGRES"
		if (g == x):
			dato = {'user':'admin','password':'root$$2018','database':'Transportes','host':'localhost'}
			conexion = mysql.connector.connect(** dato)
			cursor = conexion.cursor()
			valores = ("""INSERT INTO ingresosalida (sensor,coordenadas,dia,fecha,hora) VALUES
(%s,%s,%s,%s,%s)""",(a,b,d,e,f))
			try:
				cursor.execute(*valores)
				conexion.commit()
				print("Se ha insertado los valores dentro de la base de datos =)")
			except:
				print("No se ha podido insertar los valores dentro de la base de datos")
			conexion.close()
#-----------------------------------------------------------------
		y = "SALIDA"
		if (g == y):
			dato = {'user':'admin','password':'root$$2018','database':'Transportes','host':'localhost'}
			conexion = mysql.connector.connect(** dato)
			cursor = conexion.cursor()
			valores = ("""INSERT INTO ingresosalida (sensor,coordenadas,dia,fecha,hora) VALUES
(%s,%s,%s,%s,%s)""",(a,b,d,e,f))
			try:
				cursor.execute(*valores)
				conexion.commit()
				print("Se ha insertado los valores dentro de la base de datos =)")
			except:
```

```
                        print("No se ha podido insertar los valores dentro de la base de datos")
                conexion.close()
#-----------------------------------------------------------------
        z = "PUERTA"
        if (g == z):
                dato = {'user':'admin','password':'root$$2018','database':'Transportes','host':'localhost'}
                conexion = mysql.connector.connect(** dato)
                cursor = conexion.cursor()
                valores = ("""INSERT INTO paradas (puerta,estado,coordenadas,dia,fecha,hora) VALUES
(%s,%s,%s,%s,%s,%s)""",(a,c,b,d,e,f))
                try:
                        cursor.execute(*valores)
                        conexion.commit()
                        print("Se ha insertado los valores dentro de la base de datos =)")
                except:
                        print("No se ha podido insertar los valores dentro de la base de datos")
                conexion.close()
```

ANEXO 3. MANUAL TÉCNICO

MANUAL DE USUARIO

Control y monitoreo automático del ingreso y salida de pasajeros en un bus interprovincial con alertas SMS

Descripción.

Un control y monitoreo automático del ingreso y salida de pasajeros en un bus interprovincial con alertas SMS fue desarrollada para ayudar evitar que el operador de la unidad emita cantidades de dinero falsas o manipuladas, además de permitir saber qué es lo que está sucediendo en ese momento dentro de la unidad y notificarlo al usuario autorizado a través de un SMS o verificar el historial de los eventos acontecidos desde un navegador web.

Este prototipo, está diseñado operar dentro de cualquier unidad de transporte interprovincial, brindando así un control, seguridad, seguimiento y mejor administración del bus, el prototipo trabaja de forma automática.

Este dispositivo necesitara disponer de un paquete de mensajes contratados por cualquier operadora celular disponible en el país, para de esta manera poder dar a conocer las diferentes eventualidades que sucedan dentro de la unidad alertando al usuario autorizado a través de un SMS.

Objetivos del manual de usuario.

El objetivo principal del desarrollo del prototipo es ayudar a reducir reportes de recaudación falsos o alterados y promover el uso obligatorio del cinturón de seguridad en cualquier unidad de transporte interprovincial que lo requiera.

Componentes principales del prototipo.

En la figura 0.1, se muestra los componentes que integran el prototipo, cada uno de estos realiza un rol importante en el funcionamiento de esta.

- Batería. - Son las encargadas de suministrar la energía para el funcionamiento de sus componentes electrónicos.

- Arduino Mega 2560. - Es el encargado de controlar el funcionamiento del prototipo.
- Sensor de magnético. - El sensor verificará el momento en que la puerta se abre o se cierra.
- Módulo Laser. - El módulo laser (Transmisión y Recepción) verificará el momento en que el pasajero se encuentra ingresando o saliendo de la unidad.
- Final de carrera. - El final de carrera verificará el momento en que el pasajero se abrocha o desabrocha el cinturón de seguridad, además de indicar que butaca o asiento está disponible en ese momento.
- Módulo SIM808 GSM/GPS. - Encargado de enviar las alertas SMS al usuario autorizado de los eventos que ocurran en la unidad de transporte.
- *Display* LCD. – Encargado de informar el estatus de la cantidad de asientos libres u ocupados.
- Panel de pasajeros. - Encargado indicar que butaca o asiento en específico se encuentra ocupado o disponible y de la misma manera indica si está haciendo uso del cinturón de seguridad.
- Raspberry PI. – Almacena la base de datos del prototipo
- Regulador de voltaje. - Transforma el voltaje de entrada a un voltaje de 5V regulado para alimentar el prototipo.

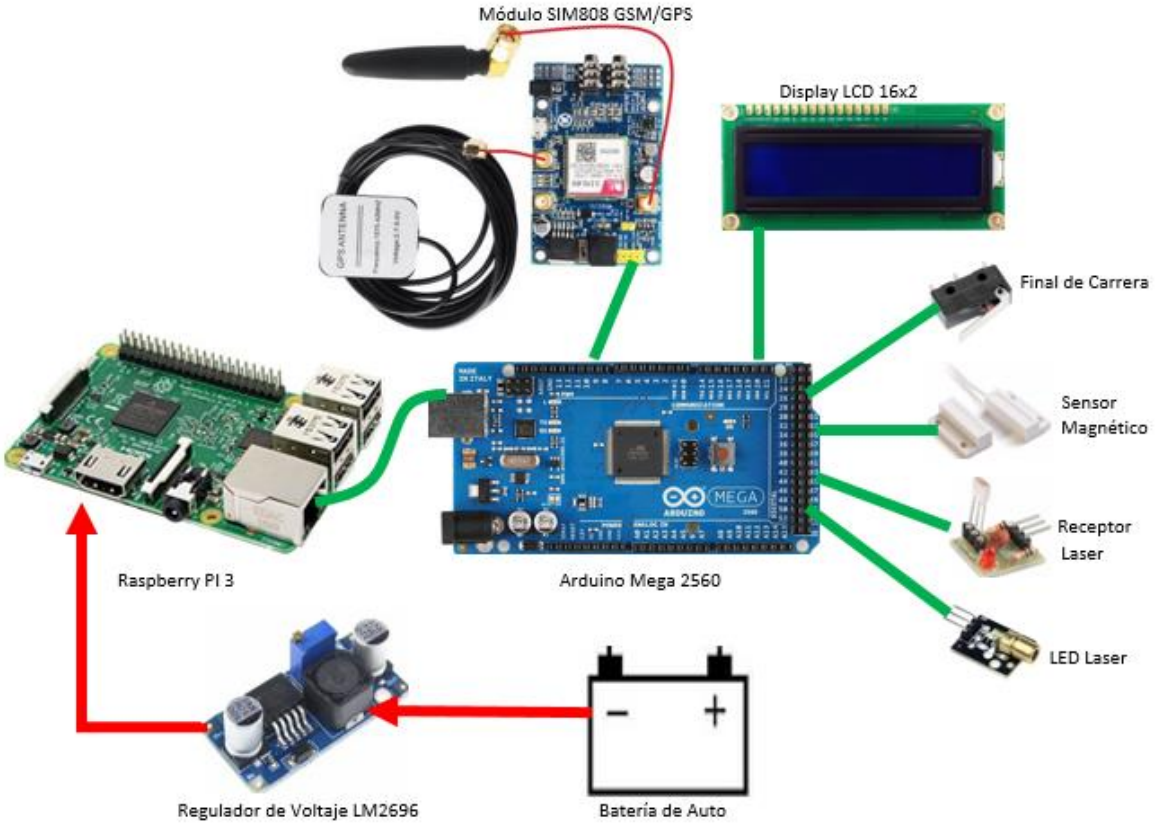


Figura. 0.1 Partes que integran el prototipo

Requerimientos para el uso del dispositivo.

- Disponer de cinturones de seguridad en cada uno de los asientos o butacas con finales de carrera instalados.
- Disponer de una tarjeta SIM con saldo para enviar las alertas.
- Tener instalado el módulo laser en la puerta del pasillo.
- Tener instalado el sensor magnético.

Montaje del final de carrera en el cinturón de seguridad.

En la siguiente figura 0.2 se observa la ubicación del final de carreta instalado en el cinturón de seguridad, de la misma manera se deberá proceder con los demás.

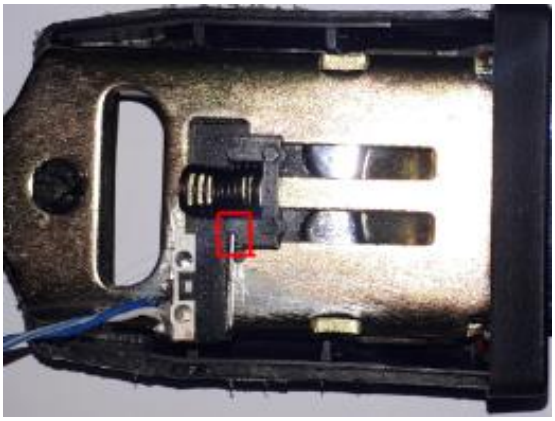


Figura. 0.2 Instalación de final de carrera

Una vez instalado el final de carrera se procederá cerrar la cubierta del cinturón de seguridad y conectar los terminales al Arduino Mega 2560.

Montaje del final de carrera en el cinturón de seguridad.

Para poder visualizar los diferentes eventos que ocurren dentro de la unidad se debe crean un usuario para poder acceder a ella. A continuación, se presentan los pasos a seguir para crear dicho usuario.

Primero se debe acceder a la página principal que se diseñó para el acceso a la base de datos desde un navegador web en la figura 0.3 se visualiza dicho enlace. Después se debe dar clic en el botón de registrase como se muestra en la figura 0.4

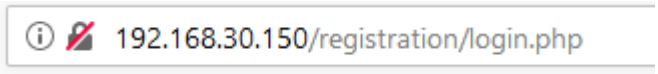


Figura. 0.3 Enlace a conectarse a la base de datos

Figura. 0.4 Botón de registrarse

Después se debe llenar todos los campos de forma obligatoria, como son usuario, correo y contraseña, ya completado este paso se debe dar clic en el botón de registrar para que se guarden los cambios. En la figura 0.5 se muestra los campos que se deben llenar.

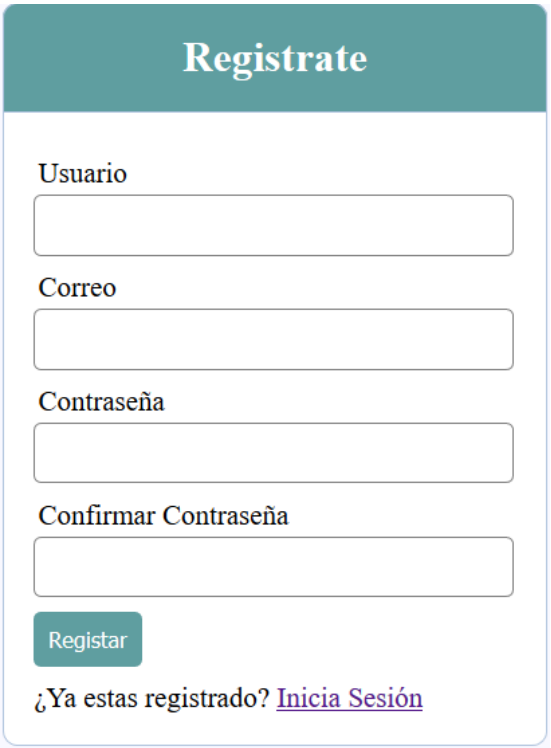


Figura. 0.5 Campos a llenarse de forma obligatoria

Luego se podrá ingresar a la base de datos con las credenciales antes creadas. En la figura 0.6 se muestra el acceso de forma exitosa a la base de datos y poder navegar libremente en ella.

Figura. 0.6 Acceso exitoso a la base de datos

Solución de problemas.

El Equipo no enciende. Verificamos si la alimentación de Raspberry PI y Módulo SIM está conectado.

El módulo SIM no arranca. Verificamos si el módulo SIM808 GSM/GPS cuenta con la tarjeta SIM valida.

No registra información en la base de datos. Verificamos que exista conectividad hacia la base de datos con un ping.

Se despliega un mensaje de error de conexión. Verificamos que el módulo SIM808 GSM/GPS se encuentre encendido.

Contacto de soporte Técnico

Correo electrónico
Rolando Javier Oña rolandoona@gmail.com

Teléfono
Rolando Oña +593 990257279

ANEXO 5. MANUAL TÉCNICO

MANUAL TÉCNICO

Control y monitoreo automático del ingreso y salida de pasajeros en un bus interprovincial con alertas SMS

Objetivos del manual de técnico.

El objetivo de este manual es dar a conocer los elementos constitutivos que conforman este prototipo y su funcionamiento, guiar a la persona encargada del mantenimiento y soporte a resolver las fallas de *hardware*, *software* más comunes y simples que se pueden presentar durante el uso del prototipo, se darán indicaciones de cómo realizar un mantenimiento preventivo y correctivo de este.

Especificaciones técnicas de los elementos constitutivos.

Arduino Mega 2560

Arduino Mega 2560, es un controlador versátil, completo y fácil de usar basado en *ATmega2560*, con más puerto de entrada y salida para uso de propósito general que el Arduino UNO, pero de diferente tamaño. Este dispone de una alimentación para voltaje DC, trabaja con un puerto USB similar al Arduino UNO. En la figura 0.7 se puede apreciar la distribución de pines del microcontrolador Arduino Mega 2560.

Fuente de poder

Arduino Mega 2560 puede ser encendido a través del puerto USB Tipo B o también desde una fuente externa que puede soportar entre 7-12 V DC y con una fuente DC en el puerto de alimentación.

Memoria.

El chip *ATmega 2560* tiene 256 KB, de los cuales 8 KB son usador para el arranque del *bootloader*, además dispone de 8 KB de memoria RAM y 4 KB de memoria EEPROM.

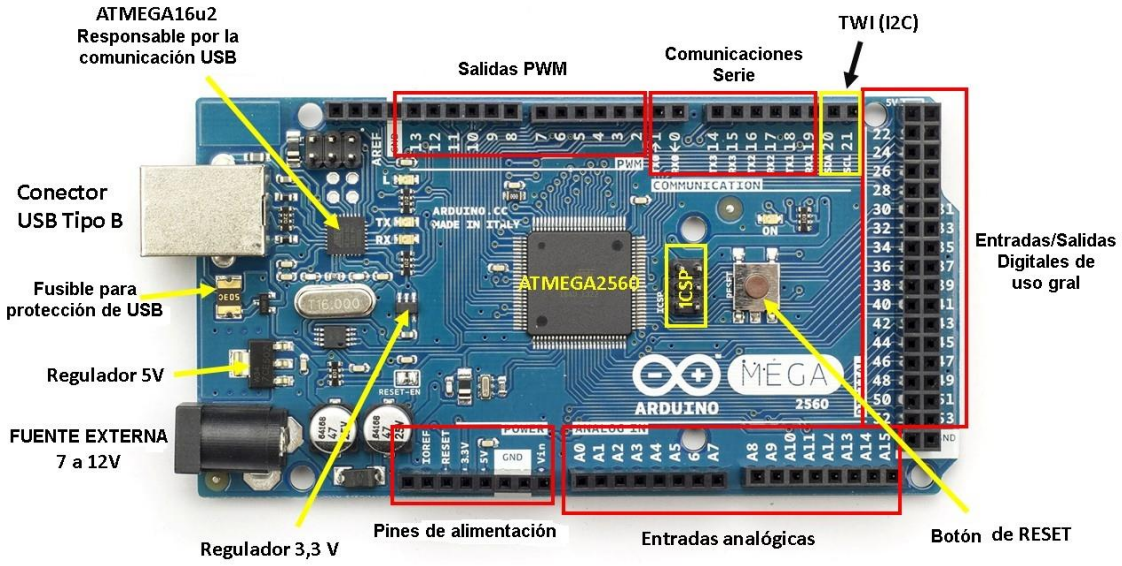


Figura. 0.7 Arduino Nano.

Entradas y salidas.

Dispone de 54 pines digitales que son usados como pines de entradas o salida, estos operan con 5V de corriente continua, cada pin puede recibir o suministrar un máximo de 40 mA, además de disponer algunos pines extras que tienen funciones especiales.

Arduino Mega 2560 dispone de 16 entradas analógicas con una resolución de 10 bits de resolución, por defecto de 0 a 5 voltios.

Comunicación.

Arduino Mega 2560 tiene la facilidad de comunicarse con un computador por medio de un puerto serial, el *ATmega16u2* provee *UART TTL* (5V) puerto serial de comunicación, está disponible en los puertos digitales (*RX) (TX*) para transmisión, el *software* de Arduino provee un puerto virtual con la computadora para recibir y enviar datos en texto plano desde el Arduino y el computador.

Programación

Los microcontroladores Arduino son programados con el *software* de Arduino IDE, que es obtenido desde la página web del fabricante, este *software* es totalmente gratuito.

Características Arduino Mega 2560.

El microcontrolador Arduino Mega 2560, tiene las siguientes características, y se mencionan en la tabla 1.

Características Arduino Mega 2560	
Microcontrolador:	*Atmega 2560*
Voltaje de alimentación de entrada:	5V
Pines digitales:	54 (15 con PWM)
Puertos analógicos:	16
Corriente máxima suministrada por pin:	40Ma
Corriente máxima suministrada para el pin 3.3V:	40 Ma
Memoria flash:	32 Kb
Memoria SRAM:	8 Kb
Memoria EEPROM:	2 kB
Velocidad de reloj:	16 MHz

Tabla 1. Características Arduino Mega 2560.

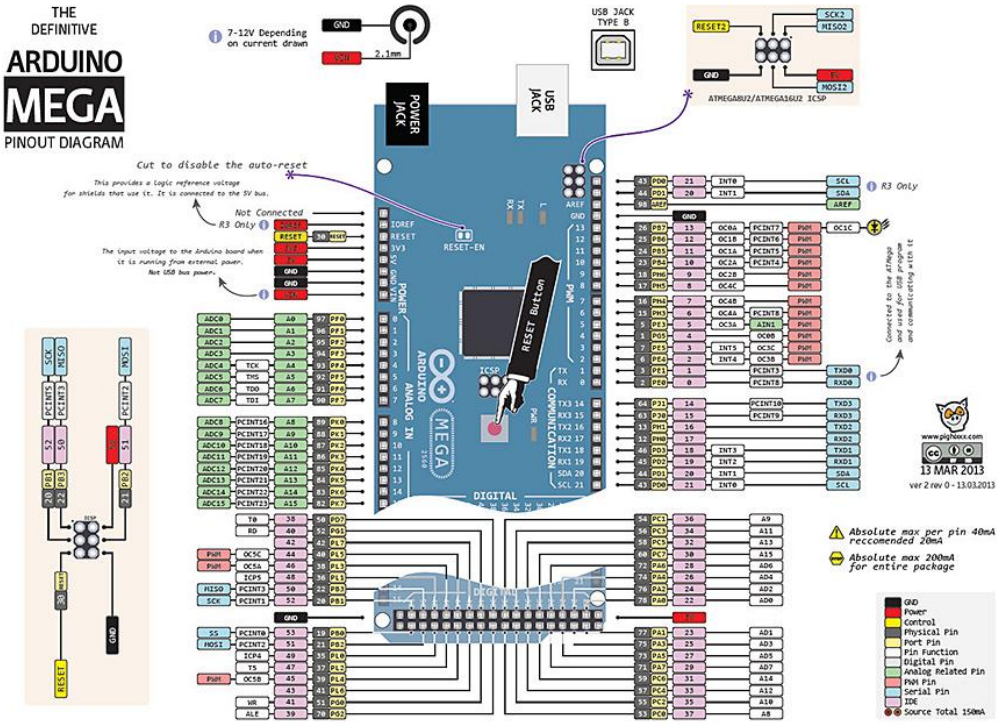


Figura. 0.8 Distribución de pines Arduino Mega 2560.

Finales de carrera.

Los interruptores finales de carrera son un interruptor (*switch*) convencional cuya única diferencia con éste último radica en dónde se lo coloca. Un final de carrera se lo coloca, justamente, al final de un desplazamiento mecánico.

Los interruptores finales de carrera constan de un accionador unido a una serie de contactos. Cuando un objeto entra en contacto con el accionador, el dispositivo activa los contactos para establecer o interrumpir una conexión eléctrica.

Están compuestos por dos partes: un cuerpo donde se encuentran los contactos y una cabeza que detecta el movimiento, en la figura 0.9 se observa la composición interna del mismo.

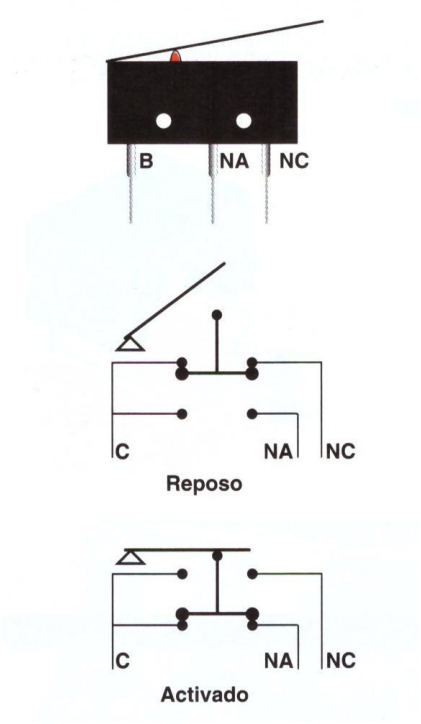


Figura. 0.9 Final de carrera.

Principales características:

- Categoría: microswitch pequeño
- No. polos: 1
- No. contactos: 2
- Configuración: C-NO-NC (común, contacto abierto, contacto cerrado)
- Numero de terminales: 3
- Largo de la palanca: 14mm
- Tipo de terminal: Terminales para soldar
- Soporte: 5A 125VAC/250VAC
- Resistencia del contacto: 30m Ohms max
- Rango de temperatura: -25ºC a +80ºC
- Vida: 100,000 ciclos

Módulo Laser KY-008

El Módulo KY-008 te permitirá realizar una infinidad de proyectos gracias al láser que se encuentra incorporado en la placa, este módulo cuenta con sólo 3 Pines de conexión para facilitar su instalación y manejo (positivo, negativo y señal).

El voltaje que requiere para funcionar oscila entre los 3.3V y 5V DC, su Potencia de funcionamiento es de 5 mW, la longitud de onda que alcanza es de 650 nm,

En algunas pruebas que hemos realizado pudimos comprobar que su alcance en condiciones adecuadas oscila entre los 1.5 y 2 Metros.

Figura. 0.10 Módulo Laser KY-008

Principales características:

- Voltaje de funcionamiento: 3.3V a 5V DC.
- Compatible con todas las tarjetas de adquisición de datos Arduino.
- Cuenta con sólo 3 pines de conexión siendo estos, positivo, negativo y el pin de la señal.
- Su potencia es de 5 mW.
- La longitud de onda es de 650 nm.
- El color de emitido es rojo.

- Posee un alcance de entre los 1.5 a 2 metros.
- El hacer bien a soldado a una placa pcb que facilita su instalación.

Módulo SIM808 GSM/GPS

SIM808 Módulo GSM GPRS GPS con antena GPS. El SIM808 es un módulo con dos funciones principales. Su diseño se creó a partir del módulo GSM / GPS SIM808 de SIMCOM. Es compatible con GSM / GPRS de cuatro bandas. Combina la tecnología GPS para obtener la posición en latitud y longitud. Su diseño incorpora un modo de consumo de baja energía y puede conectarse con sistemas de energía a base de baterías de litio. Es compatible con A-GPS. El módulo se controla mediante comandos AT mediante una interfaz de comunicación serial, puede funcionar ya sea con una lógica de voltaje de 3.3V y/o 5V. Cuando la fuente de alimentación, GSM y la antena GPS y la tarjeta SIM están conectados al módulo correctamente, el LED parpadea lentamente (3 segundo de 1 segundos la luz), que indica que el módulo está registrado en la red, y puedes hacer una llamada o hacer otra cosa, tiene una Interfaz serial TTL.

Figura. 0.11 Módulo SIM808 GSM/GPS

Principales características.

- Libre para cualquier compañía telefónica
- Compatible con tecnológica: 2G/3G/4G
- Tres entradas posibles de alimentación. V_IN, DC044 y para batería de litio.
- Entrada voltaje para DC044 y el V_IN debe ser entre 5 y 26V / 2Amp
- Entrada voltaje para la batería de Litio: de 3,5V a 4.2
- Interruptor de alimentación incluido.
- Interfaz de la antena SMA, GSM y GPS integrado BT.
- Botón de Inicio: Cuando la tarjeta se enciende, el LED (PWR) se iluminará. Después de una pulsación larga (Aproximadamente 2 segundos) los tres LED's se iluminarán. y uno de ellos comienza a parpadear, lo que indica que el SIM808 está arrancando. Cuando la fuente de alimentación, GSM y antena GPS y la tarjeta SIM está correctamente conectado al módulo, el LED se parpadeará lentamente (3 Segundos de 1 segundos la luz), que indica que el módulo está registrado en la red, y se puede hacer una llamada.
- Conexión serie TTL interfaz: una interfaz de nivel TTL (5V) tenga en cuenta que: El pasador de 3.3VMCU se utiliza para controlar el nivel de tensión alto de TTL UART.
- Interfaz USB: Esta interfaz es sólo tiene que utilizar para actualizar el firmware del módulo SIM808.

Solución de problemas más comunes.

El prototipo no enciende.

- Validar que se encuentre conectado de forma correcta.
- Verificar que los conectores USB estén conectados en el control de baterías.
- Verificar que se encuentre encendido el interruptor del módulo SIM808 GSM/GPS.

El módulo SIM808 no envía alertas SMS.

- Validar que se encuentre insertado una tarjeta SIM valida.
- Verificar que la tarjeta SIM disponga de saldo para enviar mensajes.
- Verificar que el módulo se encuentre conectado y operativo.
- Validar que el módulo SIM este recibiendo la alimentación de energía correcta.

No se puede visualizar la base de datos.

- Validar que el dispositivo (Raspberry PI) esté conectado y encendido
- Verificar con un ping al servidor de base de datos.
- Reiniciar el microcomputador.
- Validar configuración de la interfaz WIFI.

El prototipo no registra datos ni envía alertas de las butacas.

- Verificar que los cinturones de seguridad no hayan sido alterados.
- Validar que el final de carrera esté operando adecuadamente.
- Verificar la conexión física del cable serial y comunicación hacia el módulo SIM808 GSM/GPS.
- Verificar que el módulo SIM808 se haya iniciado correctamente.
- Validar LED de estatus en el microcontrolador y módulo SIM.

El prototipo se prende, pero no hace nada

- Verificar que el microcontrolador Arduino Mega 2560 no se encuentre quemado.
- Cargar nuevamente el código el microcontrolador Arduino Mega 2560.
- Verificar la script comunicación con el microcomputador.
- Verificar que el módulo SIM808 GSM/GPS no se encuentre quemado.

El prototipo ya no recibe más información dentro la base de datos.

- Verificar la comunicación serial entre el microcomputador y el microcontrolador.
- Verificar el estado de MicroSD.
- Reinstalar el sistema operativo de Raspberry PI.

Contacto de soporte Técnico

Correo electrónico
Rolando Javier Oña rolandoona@gmail.com

Teléfono
Rolando Oña +593 990257279

ANEXO 6. DIAGRAMA ELECTRÓNICO

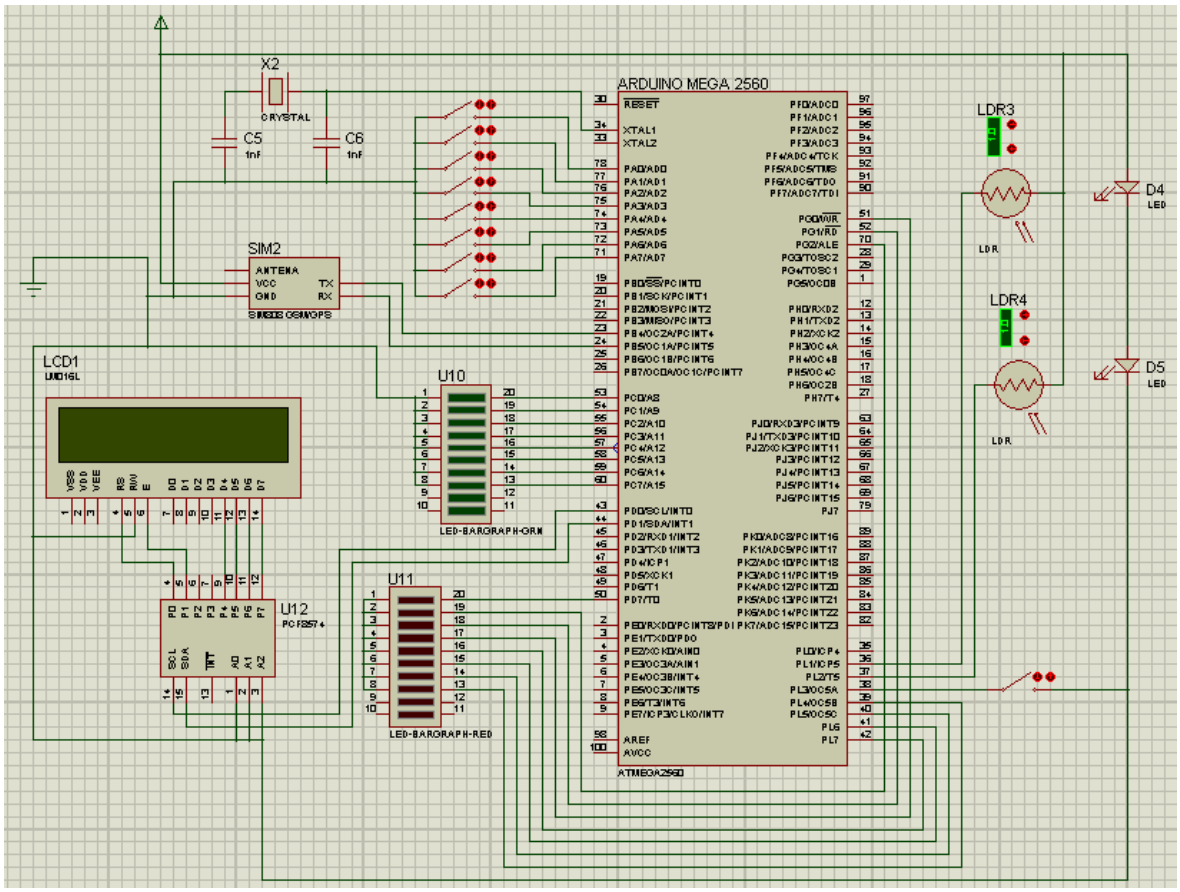
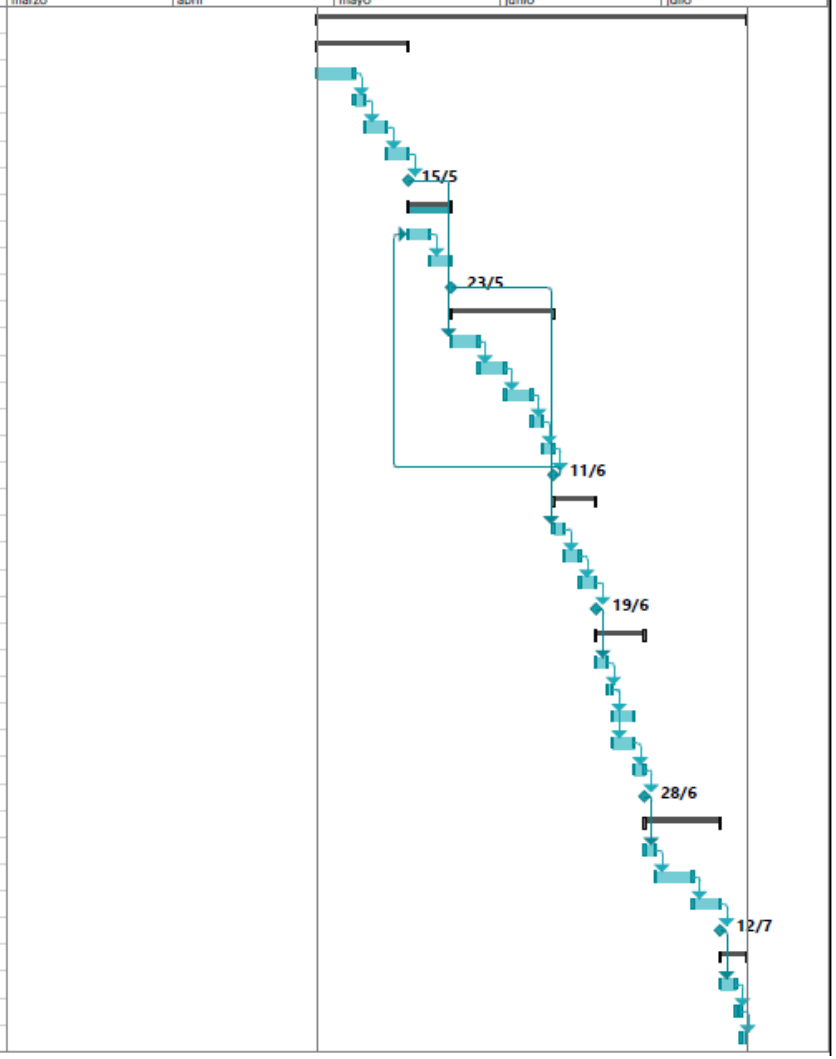


Figura. 0.12 Diagrama electrónico

ANEXO 7. CRONOGRAMA DE ACTIVIDADES

Id	Modo de tarea	Nombre de tarea	Duración	Comienzo	Fin	Nombres de los recursos
1		**PIC**	**80 días**	**sáb 28/4/18**	**lun 16/7/18**	
2		**Definir los dispositivos electrónicos para la elaboración del proyecto**	**17 días**	**sáb 28/4/18**	**mar 15/5/18**	
3		Seleccionar el tipo de sensores que se ajusten a los requerimientos	7 días	sáb 28/4/18	vie 4/5/18	
4		Delimitar la cantidad de sensores	2 días	sáb 5/5/18	dom 6/5/18	
5		Comparar las ventajas y desventajas entre los diferentes módulos GPS para geolo	4 días	lun 7/5/18	jue 10/5/18	
6		Establecer los parámetros de compatibilidad que tienen los diferentes módulos SI	4 días	vie 11/5/18	lun 14/5/18	
7		Fin de la definición de los dispositivos electrónicos para la elaboración del proyect	0 días	mar 15/5/18	mar 15/5/18	
8		**Diseñar los diagramas necesarios para el control y monitoreo de transporte de pas**	**8 días**	**mar 15/5/18**	**mar 22/5/18**	
9		Realizar el diagrama de conexión de los diferentes dispositivos electronicos	4 días	mar 15/5/18	vie 18/5/18	
10		Determinar el lugar adecuado de los dispositivos electronicos usados	4 días	sáb 19/5/18	mar 22/5/18	
11		Fin del diseño de los diagramas necesarios para el control y monitoreo de transpo	0 días	mié 23/5/18	mié 23/5/18	
12		**Definir la programación respectiva en el microcontrolador Arduino**	**19 días**	**mié 23/5/18**	**lun 11/6/18**	
13		Realizar el código de programación en Arduino con los sensores adecuados	5 días	mié 23/5/18	dom 27/5/18	
14		Realizar el código de programación respectiva con el modulo GPS adquirido para	5 días	lun 28/5/18	vie 1/6/18	
15		Realizar el código de programación respectiva con el modulo GSM adquirido para	5 días	sáb 2/6/18	mié 6/6/18	
16		Unir los códigos de programación realizados y realizar pruebas de funcionamient	2 días	jue 7/6/18	vie 8/6/18	
17		Depurar posibles inconvenientes en el código de programación.	2 días	sáb 9/6/18	dom 10/6/18	
18		Fin de la definición de la programación repespectiva en el microcontrolador Ardín	0 días	lun 11/6/18	lun 11/6/18	
19		**Realizar la comunicación entre Arduino y Raspberry PI**	**8 días**	**lun 11/6/18**	**mar 19/6/18**	
20		Instalar el sistema operativo en el micro ordenador Raspberry PI	2 días	lun 11/6/18	mar 12/6/18	
21		Desarrollar el código de programación en Python para leer datos que se registran	3 días	mié 13/6/18	vie 15/6/18	
22		Realizar el código de programación para que los datos provenientes de la comuni	3 días	sáb 16/6/18	lun 18/6/18	
23		Fin de la realización de la comunicación entre Arduino y Raspberry PI	0 días	mar 19/6/18	mar 19/6/18	
24		**Crear la base de datos**	**9 días**	**mar 19/6/18**	**jue 28/6/18**	
25		Instalar el Servidor APACHE, PHP, PhpMyAdmin y MySQL en la Raspberry PI	2 días	mar 19/6/18	mié 20/6/18	
26		Verificar que APACHE, PHP, PhpMyAdmin y MSQL se hayan instalado correctame	1 día	jue 21/6/18	jue 21/6/18	
27		Crear las tablas en el programa MySQL	4 días	vie 22/6/18	lun 25/6/18	
28		Realizar las relaciones correspondientes con cada tabla	4 días	vie 22/6/18	lun 25/6/18	
29		Exportar la base de datos creada a PhpMyAdmin	2 días	mar 26/6/18	mié 27/6/18	
30		Fin de la creación de la base de datos	0 días	jue 28/6/18	jue 28/6/18	
31		**Implemetación del prototipo en un modelo o maqueta**	**14 días**	**jue 28/6/18**	**jue 12/7/18**	
32		Seleccionar el material adecuado para realizar el modelo o maqueta del prototipo	2 días	jue 28/6/18	vie 29/6/18	
33		Construir la maqueta del prototipo	7 días	sáb 30/6/18	vie 6/7/18	
34		Instalar el hardware en la maqueta para demostar el funcionamiento del prototip	5 días	sáb 7/7/18	mié 11/7/18	
35		Fin de la implemetación del prototipo en un modelo o maqueta	0 días	jue 12/7/18	jue 12/7/18	
36		**Ejecutar pruebas de funcionamiento**	**5 días**	**jue 12/7/18**	**lun 16/7/18**	
37		Realizar pruebas de prototipo.	3 días	jue 12/7/18	sáb 14/7/18	
38		Detectar problemas de funconamiento.	1 día	dom 15/7/18	dom 15/7/18	
39		Corregir fallos de operación del prototipo	1 día	lun 16/7/18	lun 16/7/18	



DECLARACIÓN Y AUTORIZACIÓN

Yo, Rolando Javier Oña Oña, CI 0503925877 autor/a del trabajo de graduación:

Desarrollo de un prototipo para el control y monitoreo automático del ingreso y salida de pasajeros en un bus interprovincial con alertas SMS, previo a la obtención del título de **Ingeniería Electrónica Digital y Telecomunicaciones** en la UNIVERSIDAD TECNOLÓGICA ISRAEL.

1.- Declaro tener pleno conocimiento de la obligación que tienen las instituciones de educación superior, de conformidad con el Articulo 144 de la Ley Orgánica de Educación Superior, de difundir el respectivo trabajo de graduación para que sea integrado al Sistema Nacional de Información de la Educación Superior del Ecuador para su difusión pública respetando los derechos de autor.

2.- Autorizo a la SENESCYT a tener una copia del referido trabajo de graduación, con el propósito de generar un repositorio que democratice la información, respetando las políticas de propiedad intelectual vigentes.

Quito, Viernes 12 de Julio del 2019

Atentamente.

Rolando Javier Oña Oña.

C.I. 0503925877