Universidad de San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Ingeniería en Ciencias y Sistemas

SISTEMA DE ADMINISTRACIÓN DE SERVICIOS Y EMERGENCIAS PARA EL BENEMÉRITO CUERPO DE BOMBEROS VOLUNTARIOS DE GUATEMALA

Rolando Giovanni Marroquín González
Asesorado por el Ingeniero Shai Alberto Chilin Bolaños

Guatemala, mayo de 2023

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA

FACULTAD DE INGENIERÍA

SISTEMA DE ADMINISTRACIÓN DE SERVICIOS Y EMERGENCIAS PARA EL BENEMÉRITO CUERPO DE BOMBEROS VOLUNTARIOS DE GUATEMALA

TRABAJO DE GRADUACIÓN

PRESENTADO A LA JUNTA DIRECTIVA DE LA
FACULTAD DE INGENIERÍA
POR

ROLANDO GIOVANNI MARROQUÍN GONZÁLEZ
ASESORADO POR EL INGENIERO SHAI ALBERTO CHILIN BOLAÑOS

AL CONFERÍRSELE EL TÍTULO DE

INGENIERO EN CIENCIAS Y SISTEMAS

GUATEMALA, MAYO DE 2023

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA

FACULTAD DE INGENIERÍA

NÓMINA DE JUNTA DIRECTIVA

DECANA	Inga. Aurelia Anabela Cordova Estrada
VOCAL I	Ing. José Francisco Gómez Rivera
VOCAL II	Ing. Mario Renato Escobedo Martínez
VOCAL III	Ing. José Milton de León Bran
VOCAL IV	Br. Kevin Vladimir Cruz Lorente
VOCAL V	Br. Fernando José Paz González
SECRETARIO	Ing. Hugo Humberto Rivera Pérez

TRIBUNAL QUE PRACTICÓ EL EXAMEN GENERAL PRIVADO

DECANA	Inga. Aurelia Anabela Cordova Estrada
EXAMINADOR	Ing. César Augusto Fernández Cáceres
EXAMINADOR	Ing. Juan Álvaro Días Ardavín
EXAMINADOR	Ing. Álvaro Giovanni Longo Morales
SECRETARIO	Ing. Hugo Humberto Rivera Pérez

HONORABLE TRIBUNAL EXAMINADOR

En cumplimiento con los preceptos que establece la ley de la Universidad de San Carlos de Guatemala, presento a su consideración mi trabajo de graduación titulado:

SISTEMA DE ADMINISTRACIÓN DE SERVICIOS Y EMERGENCIAS PARA EL BENEMÉRITO CUERPO DE BOMBEROS VOLUNTARIOS DE GUATEMALA

Tema que me fuera asignado por la Dirección de la Escuela de Ingeniería en Ciencias y Sistemas, con fecha 23 de agosto de 2021.

Rolando Giovanni Marroquín González

Guatemala, 13 de marzo de 2023

Ingeniero
Carlos Alfredo Azurdia
Coordinador de Privados y Trabajos de Tesis
Escuela de Ingeniería en Ciencias y Sistemas
Facultad de Ingeniería - USAC

Respetable Ingeniero Azurdia:

Por este medio hago de su conocimiento que en mi rol de asesor del trabajo de investigación realizado por el estudiante **ROLANDO GIOVANNI MARROQUÍN GONZÁLEZ** con carné **200312642 y CUI 1706 33764 0114** titulado **"SISTEMA DE ADMINISTRACIÓN DE SERVICIOS Y EMERGENCIAS PARA EL BENEMERITO CUERPO DE BOMBEROS VOLUNTARIOS DE GUATEMALA",** luego de corroborar que el mismo se encuentra finalizado, lo he revisado y doy fé de que el mismo cumple con los objetivos propuestos en el respectivo protocolo, por consiguiente, procedo a la aprobación correspondiente.

Al agradecer su atención a la presente, aprovecho la oportunidad para suscribirme,

Atentamente,

Ing. Shai Alberto Chilin Bolaños
Colegiado No. 15334

Universidad San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Ingeniería en Ciencias y Sistemas

Guatemala 20 de marzo de 2023

Ingeniero
Carlos Gustavo Alonzo
Director de la Escuela de Ingeniería
En Ciencias y Sistemas

Respetable Ingeniero Alonzo:

Por este medio hago de su conocimiento que he revisado el trabajo de graduación del estudiante **ROLANDO GIOVANNI MARROQUÍN GONZÁLEZ** con carné **200312642** y CUI **1706 33764 0114** titulado **"SISTEMA DE ADMINISTRACIÓN DE SERVICIOS Y EMERGENCIAS PARA EL BENEMERITO CUERPO DE BOMBEROS VOLUNTARIOS DE GUATEMALA",** y a mi criterio el mismo cumple con los objetivos propuestos para su desarrollo, según el protocolo aprobado.

Al agradecer su atención a la presente, aprovecho la oportunidad para suscribirme,

Atentamente,

Ing. Carlos Alfredo Azurdia
Coordinador de Privados
y Revisión de Trabajos de Graduación

UNIVERSIDAD DE SAN CARLOS
DE GUATEMALA

FACULTAD DE INGENIERÍA

LNG.DIRECTOR.107.EICCSS.2023

El Director de la Escuela de Ingeniería en Ciencias y Sistemas de la Facultad de Ingeniería de la Universidad de San Carlos de Guatemala, luego de conocer el dictamen del Asesor, el visto bueno del Coordinador de área y la aprobación del área de lingüística del trabajo de graduación titulado: **SISTEMA DE ADMINISTRACIÓN DE SERVICIOS Y EMERGENCIAS PARA EL BENEMÉRITO CUERPO DE BOMBEROS VOLUNTARIOS DE GUATEMALA**, presentado por: **Rolando Giovanni Marroquín González** , procedo con el Aval del mismo, ya que cumple con los requisitos normados por la Facultad de Ingeniería.

"ID Y ENSEÑAD A TODOS"

Ing. Carlos Gustavo Alonzo
Director
Escuela de Ingeniería en Ciencias y Sistemas

MSc. Ing. Carlos Gustavo Alonzo
Director
Escuela de Ingeniería en Ciencias y Sistemas

Guatemala, mayo de 2023

Decanato
Facultad de Ingeniería
24189101- 24189102
secretariadecanato@ingenieria.usac.edu.gt

LNG.DECANATO.OI.425.2023

La Decana de la Facultad de Ingeniería de la Universidad de San Carlos de Guatemala, luego de conocer la aprobación por parte del Director de la Escuela de Ingeniería en Ciencias y Sistemas, al Trabajo de Graduación titulado: **SISTEMA DE ADMINISTRACIÓN DE SERVICIOS Y EMERGENCIAS PARA EL BENEMÉRITO CUERPO DE BOMBEROS VOLUNTARIOS DE GUATEMALA,** presentado por: **Rolando Giovanni Marroquín González,** después de haber culminado las revisiones previas bajo la responsabilidad de las instancias correspondientes, autoriza la impresión del mismo.

IMPRÍMASE:

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA
DECANA
FACULTAD DE INGENIERÍA

Inga. Aurelia Anabela Cordova Estrada
Decana

Guatemala, mayo de 2023

AACE/gaoc

ACTO QUE DEDICO A:

Dios	Por guiarme en todo este proceso, por siempre estar presente en cada paso que doy y por iluminar mi carrera profesional.
Mis padres	Martín Roberto de León y Xiomara Lucrecia González, por todo su amor y sacrificio con el que me hicieron ser un profesional de bien e inculcarme los valores que hoy hacen posible este logro.
Mi esposa	Miriam Del Rosario Carballo Almeda de Marroquín, por todo su amor, por siempre ser la persona que ilumina mi vida, por ser la inspiración para nunca desmayar, por ser mi compañera de vida.
Mi hija	Andrea Michelle Marroquín Carballo, por permitirme ser su padre, por darme el ejemplo y enseñarme el camino que debo seguir.
Mis hermanos	Carlos Francisco Marroquín y María Joseé De León. Por ser dos pilares en mi vida, por darme el ejemplo como profesionales de bien, por siempre estar para mí en los momentos más difíciles y por el amor que nos une.

Mis abuelos Carlos Rolando González López y Zoila Esperanza Contreras de González (q. e. p. d), por ser un ejemplo de vida y sacrificio. Por darme una infancia maravillosa, por estar siempre al pendiente de mí, por nunca dejarme caminar solo y por todo el amor con el que me hicieron ser lo que hoy puedo ser, gracias por estar a mi lado en todo momento.

Mis suegros Jorge Abel Carballo Aldana (q. e. p. d) y Berta Almeda López De Carballo, por abrirme las puertas de su familia, por su cariño, por confiar a su mayor tesoro en mí y por siempre apoyarme en todo momento.

Mi nieta Sasha María Marroquín Carballo (q. e. p. d), por ser mi fiel compañera en cada momento, por enseñarme a no desmayar incluso en los momentos más difíciles de la vida, por el amor incondicional que nos unió y transcendió a la eternidad. Gracias por escogernos como tu familia terrenal.

AGRADECIMIENTOS A:

Universidad de San Carlos de Guatemala	Por albergar mis sueños durante los años que duro mi preparación profesional, por darme el poder del conocimiento y hacerme libre, mi alma máter muchas gracias.
Facultad de Ingeniería	Por hacer de mi un profesional de honor, de bien y por darme las herramientas para ayudar a engrandecer a mi país a través de los conocimientos adquiridos.
Mi asesor	Shai Alberto Chilin Bolaños, por ayudarme en la culminación de este sueño, por estar siempre dispuesto a colaborar con mi persona.
Mi cuñado	José Daniel Chilin Bolaños, por encontrar una excelente persona en quien confiar mi familia, por darme su ayuda incondicional.
Mis profesores	Por transmitirme todos sus conocimientos, sus experiencias profesionales, gracias a sus enseñanzas podré continuar su legado profesional.

ÍNDICE GENERAL

ÍNDICE DE ILUSTRACIONES

FIGURAS

TABLAS

GLOSARIO

API	*Aplication Programming Interface*, conjunto de rutinas de *software* que pueden ser utilizados como una librería desde otro *software* por medio de invocaciones remotas.
Chekout	Proceso de revisar los artículos que se compraran y realizar la compra en el sitio web.
FrameWork	Entorno de trabajo, conjunto estandarizado de herramientas y reglas para desarrollar un programa de *software*.
HTML	Lenguaje de marcas de hiper texto, es el lenguaje que interpretan los navegadores web para poder interpretar la información de los componentes y su distribución en una página web.
HTTP	Protocolo de transferencia de hiper texto. Principal protocolo de comunicación en internet.
JavaScript	Lenguaje de programación interpretado, orientado al entorno web.
JSON	Formato de texto para el intercambio de datos, basado en la notación de objetos de JavaScript.

MIT	Instituto de tecnología de Massachusetts.
Nube	Se refiere a todo lo contenido en internet, no es una entidad física, es una red de servidores que se ven como un único ente, la internet en su totalidad.
PDF	Formato de documento portable, utilizado para almacenar documentos en formato digital independiente de la plataforma.
PHP	Lenguaje de programación de propósito general especializado al entorno web.
Plugin	Complemento a programas o aplicaciones que agregan funcionalidad.
Streaming	Tecnología que permite transmitir contenidos multimedia en tiempo real, sin necesidad de descargarlos previamente.
Web	Concepto utilizado para referirse a una red informática, en general es usado para referirse al internet.
XML	Lenguaje de marca extensible, lenguaje utilizado para la codificación de documentos, utilizado en el intercambio de datos.

RESUMEN

El sistema de administración de servicios y emergencias es una aplicación web en la cual se podrá llevar el control de la operación de una compañía de Bomberos Voluntarios. Las compañías que se encuentran tanto en la capital como en el interior de la república podrán disponer de un sistema para llevar el control de forma digital de los servicios que prestan a sus comunidades, las emergencias que atienden, así como manejar los catálogos de unidades disponibles, herramientas, equipo, insumos, bomberos activos, control de las llamadas de emergencias recibidas en cabina, las unidades que se destacan para atender dichas emergencias y reportes de las emergencias atendidas.. El sistema se podrá acondicionar a las necesidades administrativas básicas de una estación de Bomberos Voluntarios en cualquier parte de la república.

La aplicación se publicará en un dominio .com, con el objetivo de estar disponible desde cualquier dispositivo, en todo momento. Contará con una página de presentación como inicio, en la cual se podrá leer información pública de la compañía y generalidades que se quieran dar a conocer. Dentro de esta página estará el enlace para dirigirse al sistema de administración interno, el cual estará protegido por medio de usuario y contraseña.

El principal objetivo de la aplicación es que la compañía de bomberos donde se utilice pueda automatizar las tareas básicas en la operación de la administración de servicios prestados por esta institución.

OBJETIVOS

General

Proveer al benemérito cuerpo de Bomberos Voluntarios los medios tecnológicos para una administración efectiva y moderna de sus estaciones a través de una aplicación que les permita llevar el control de sus operaciones diarias, control de sus principales insumos, herramientas, recurso humano y vehículos. Principalmente eliminar el uso de formatos de papel para llevar el control de sus operaciones. Además de poder atender los requerimientos de entidades oficiales en cuanto a información histórica de emergencias atendidas se refiere, ya que esta es de vital importancia en las áreas donde se presta el servicio.

Específicos

1. Crear el control de llamadas de emergencia recibidas en cabina.

2. Crear el control de servicios prestados por la compañía.

3. Crear el control de bomberos dados de alta y baja.

4. Desarrollar cada una de las pantallas y procesos de forma intuitiva, moderna y eficiente.

5. Publicar la aplicación en internet para acceso desde cualquier lugar y dispositivo.

INTRODUCCIÓN

El benemérito cuerpo de Bomberos Voluntarios de Guatemala es una entidad que presta el servicio de atención de emergencias en todo el país de manera ininterrumpida los 365 días del año. Esta institución se maneja a base de donaciones y un presupuesto muy limitado, por tal razón sus recursos siempre han sido muy reducidos, dentro de estos, como es de suponerse, los recursos tecnológicos son muy escasos, haciéndose más evidente en las estaciones del interior del país.

Por lo antes expuesto, la necesidad de automatizar algunas de las tareas que requieren la operación de una estación es de vital importancia. Este trabajo de tesis parte de esta premisa, proveer a esta noble institución de una herramienta tecnológica para poder automatizar los procesos manuales que se llevan a cabo hoy en día.

El sistema de administración de servicios y emergencias como primera versión estará basada en los procesos de administración básicos de la Vigésimo Novena Compañía de Bomberos Voluntarios de Amatitlán, ya que sus procesos son los más genéricos, si bien es cierto que cada compañía de Bomberos Voluntarios es independiente y tienen sus peculiaridades administrativas, se iniciara con el modelo propuesto por la mencionada compañía de bomberos.

El sistema únicamente se encargará de la administración interna de la compañía de Bomberos Voluntarios con sede en el municipio de Amatitlán del departamento de Guatemala, es decir, únicamente se automatizaran los procesos administrativos internos de dicha estación, se deja fuera del alcance de

este trabajo la forma en la que la población reporta las emergencias, el sistema para reportar las llamadas de emergencia que se utiliza actualmente seguirá siendo el mismo, por medio del número telefónico 122 a nivel nacional y 66330333 el cual corresponde al número telefónico local, no se implementará ningún modulo para cambiar este proceso.

La visión que se tiene es que este sistema sea el punto de partida de un sistema integral a nivel nacional en el cual se integren las demás compañías de Bomberos Voluntarios a mediano y largo plazo, es decir, sentar las bases para la modernización de esta institución y la noble labor que incansablemente realizan en beneficio de todos los guatemaltecos.

1. MARCO TEÓRICO

Los Bomberos Voluntarios de Guatemala son una entidad no lucrativa que se dedica a la atención de emergencias en todo el país, dicha entidad tiene un modelo de estaciones independientes que se ubican tanto en la capital como en el interior del país. Cada una de estas estaciones presta diferentes servicios en las comunidades donde están ubicadas.

La vigésimo novena compañía de Bomberos Voluntarios está ubicada en el municipio de Amatitlán, Guatemala. Esta compañía de bomberos presta el servicio de atención de emergencias a todo el municipio, cuenta con una motobomba para la atención de incendios y 3 unidades de rescate con las cuales realizan la atención de primeros auxilios y traslado de personas accidentadas o enfermas al hospital de la localidad.

A continuación, se describe la situación actual de sus procesos administrativos, así como los conceptos clave sobre las tecnologías en las cuales estará basada la herramienta de administración propuesta con la cual se logrará la automatización de dichos procesos.

1.1. Problemática actual

Actualmente dentro de la vigésimo novena compañía de Bomberos Voluntarios de Amatitlán se realizan todos los procesos administrativos sobre papel, algunas operaciones se han logrado implementar en hojas electrónicas, sin embargo, esto ha dificultado la tabulación de la información histórica, se depende aun de procesos manuales que son propensos al error humano, lo cual

se traduce en información histórica inexacta. Además del error humano se tiene la dificultad de no contar con los equipos adecuados para resguardar la poca información que se logra digitalizar. En años anteriores se ha tenido el inconveniente de perder información histórica en equipos que por su obsolescencia y antigüedad dejan de funcionar, por ende, los medios magnéticos donde se ha almacenado esta información quedan inservibles, perdiéndose así información de vital importancia para la estación y el municipio.

Una de las grandes necesidades de resguardar esta información histórica, por mencionar algunas, es el reporte que se genera por parte de los bomberos en las emergencias atendidas, en ciertas ocasiones este reporte es requerido por las entidades de justicia, ya que los incidentes registrados en este documento forman parte de procesos penales o judiciales cuando se daña a terceros, propiedad privada, entre otros. Además, la información histórica de las emergencias atendidas también son un medidor plausible para las autoridades municipales locales, ya que con base en estas estadísticas presentadas se pueden gestionar donaciones de insumos, herramientas y vehículos que son necesarios para atender a la población del municipio.

Los formatos de papel en los cuales se registra la información de emergencias atendidas dificultan y ralentizan la generación de dichos reportes, además de no contar con el sistema adecuado para almacenarlos, estos sufren deterioro por condiciones climáticas y mal manejo, lo cual conlleva a la perdida de información. Tabular información que proviene de formatos impresos conlleva una labor titánica para poder generar alguna estadística o simplemente verificar patrones de áreas en donde se repiten siniestros y accidentes con mayor frecuencia.

1.2. Registro de información

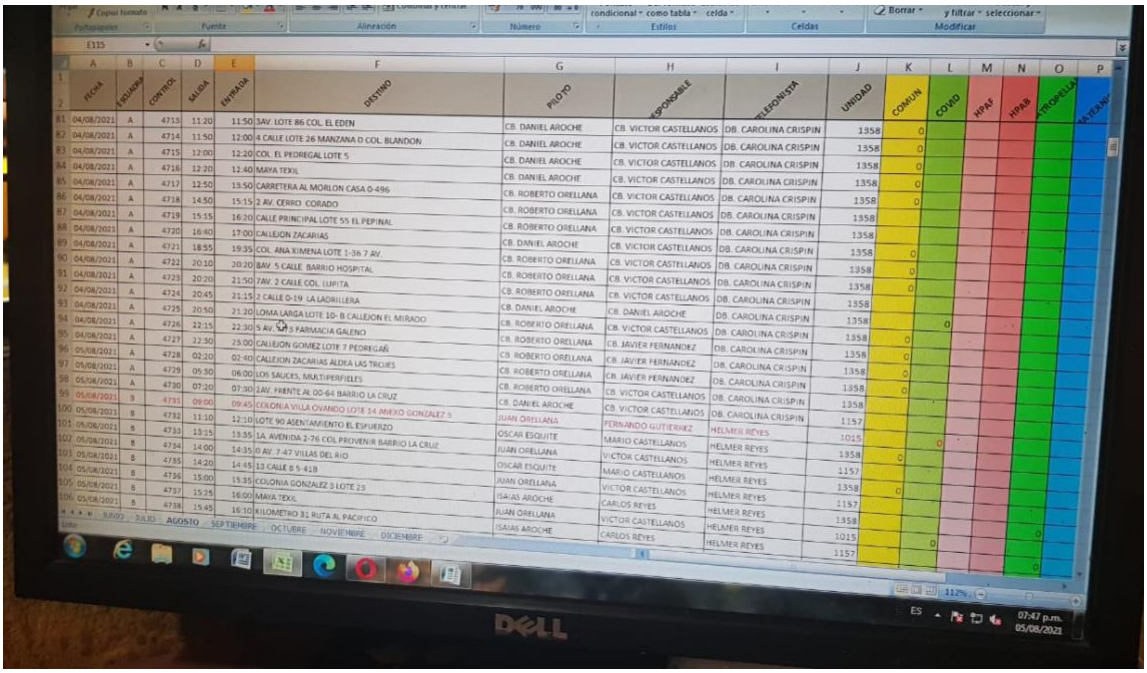
Cuando la cabina de bomberos recibe una llamada solicitando atención a una emergencia o servicio, se inicia con el procedimiento de registro de la información, este proceso se describe a continuación.

1.2.1. Registro de llamadas

Actualmente se cuenta con un formato de papel para el registro de las llamadas de emergencia o de solicitud de servicios varios. En este formato o boleta se anota el correlativo de la llamada, lo cual es control manual que se reinicia cada año, la dirección a donde se debe ir a atender la emergencia o servicio, el teléfono de donde se está reportando la emergencia, el nombre de la persona de quien reporta la emergencia, el tipo de emergencia o servicio que se está solicitando, la unidad que se asigna, fecha de la solicitud, hora de salida de la unidad de rescate, hora de entrada de la unidad de rescate, piloto de la unidad de rescate, responsable en la estación de bomberos en ese turno, telefonista que atendió la llamada de emergencia, así como el telefonista designado como el telefonista de central en ese turno.

La información capturada en la boleta de solicitud de servicio posteriormente se traslada hacia una hoja electrónica con los datos básicos de cada llamada. Este será de aquí en adelante el único registro digital que se tendrá de la llamada. Las boletas físicas son almacenadas en cajas de cartón o recipientes que estén a mano para posteriormente ser trasladados a la bodega que se encuentra en la estación.

Figura 1. **Hoja electrónica de llamadas**



	FECHA	EQUIPO	CONTROL	SALIDA	ENTRADA	DESTINO	PILOTO	RESPONSABLE	TELEFONISTA	UNIDAD	COMUN	COVID	HPAF	HPAB	TROPELLA	TETANO
81	04/08/2021	A	4713	11:20	11:50	3AV. LOTE 86 COL. EL EDEN	CB. DANIEL AROCHE	CB. VICTOR CASTELLANOS	DB. CAROLINA CRISPIN	1358	0					
82	04/08/2021	A	4714	11:50	12:00	4 CALLE LOTE 26 MANZANA O COL. BLANDON	CB. DANIEL AROCHE	CB. VICTOR CASTELLANOS	DB. CAROLINA CRISPIN	1358	0					
83	04/08/2021	A	4715	12:00	12:20	COL. EL PEDREGAL LOTE 5	CB. DANIEL AROCHE	CB. VICTOR CASTELLANOS	DB. CAROLINA CRISPIN	1358	0					
84	04/08/2021	A	4716	12:20	12:40	MAYA TEXIL	CB. DANIEL AROCHE	CB. VICTOR CASTELLANOS	DB. CAROLINA CRISPIN	1358	0					
85	04/08/2021	A	4717	12:50	13:50	CARRETERA AL MORLON CASA 0-496	CB. ROBERTO ORELLANA	CB. VICTOR CASTELLANOS	DB. CAROLINA CRISPIN	1358	0					
86	04/08/2021	A	4718	14:50	15:15	2 AV. CERRO CORADO	CB. ROBERTO ORELLANA	CB. VICTOR CASTELLANOS	DB. CAROLINA CRISPIN	1358	0					
87	04/08/2021	A	4719	15:15	16:20	CALLE PRINCIPAL LOTE 55 EL PEPINAL	CB. ROBERTO ORELLANA	CB. VICTOR CASTELLANOS	DB. CAROLINA CRISPIN	1358						
88	04/08/2021	A	4720	16:40	17:00	CALLEJON ZACARIAS	CB. DANIEL AROCHE	CB. VICTOR CASTELLANOS	DB. CAROLINA CRISPIN	1358						
89	04/08/2021	A	4721	18:55	19:35	COL. ANA XIMENA LOTE 1-36 7 AV.	CB. ROBERTO ORELLANA	CB. VICTOR CASTELLANOS	DB. CAROLINA CRISPIN	1358	0					
90	04/08/2021	A	4722	20:10	20:20	8AV. 5 CALLE BARRIO HOSPITAL	CB. ROBERTO ORELLANA	CB. VICTOR CASTELLANOS	DB. CAROLINA CRISPIN	1358	0					
91	04/08/2021	A	4723	20:20	21:50	7AV. 2 CALLE COL. LUPITA	CB. ROBERTO ORELLANA	CB. VICTOR CASTELLANOS	DB. CAROLINA CRISPIN	1358	0					
92	04/08/2021	A	4724	20:45	21:15	2 CALLE 0-19 LA LADRILLERA	CB. ROBERTO ORELLANA	CB. VICTOR CASTELLANOS	DB. CAROLINA CRISPIN	1358						
93	04/08/2021	A	4725	20:50	21:20	LOMA LARGA LOTE 10- B CALLEJON EL MIRADO	CB. DANIEL AROCHE	CB. DANIEL AROCHE	DB. CAROLINA CRISPIN	1358						
94	04/08/2021	A	4726	22:15	22:30	5 AV. 5-35 FARMACIA GALENO	CB. ROBERTO ORELLANA	CB. VICTOR CASTELLANOS	DB. CAROLINA CRISPIN	1358		0				
95	04/08/2021	A	4727	22:30	23:00	CALLEJON GOMEZ LOTE 7 PEDREGAL	CB. ROBERTO ORELLANA	CB. JAVIER FERNANDEZ	DB. CAROLINA CRISPIN	1358	0					
96	05/08/2021	A	4728	02:20	02:40	CALLEJON ZACARIAS ALDEA LAS TROJES	CB. ROBERTO ORELLANA	CB. JAVIER FERNANDEZ	DB. CAROLINA CRISPIN	1358	0					
97	05/08/2021	A	4729	05:30	06:00	LOS SAUCES, MULTIPERFILES	CB. ROBERTO ORELLANA	CB. JAVIER FERNANDEZ	DB. CAROLINA CRISPIN	1358	0					
98	05/08/2021	A	4730	07:20	07:30	2AV. FRENTE AL 00-64 BARRIO LA CRUZ	CB. ROBERTO ORELLANA	CB. VICTOR CASTELLANOS	DB. CAROLINA CRISPIN	1358	0					
99	05/08/2021	B	4731	09:00	09:45	COLONIA VILLA OVANDO LOTE 14 ANEXO GONZALEZ 3	CB. DANIEL AROCHE	CB. VICTOR CASTELLANOS	DB. CAROLINA CRISPIN	1358						
100	05/08/2021	B	4732	11:10	12:10	LOTE 90 ASENTAMIENTO EL ESFUERZO	JUAN ORELLANA	FERNANDO GUTIERREZ	HELMER REYES	1157						
101	05/08/2021	B	4733	13:15	13:55	1A AVENIDA 2-76 COL PROVENIR BARRIO LA CRUZ	OSCAR ESQUITE	MARIO CASTELLANOS	HELMER REYES	1015		0				
102	05/08/2021	B	4734	14:00	14:35	0 AV. 7-47 VILLAS DEL RIO	JUAN ORELLANA	VICTOR CASTELLANOS	HELMER REYES	1358	0					
103	05/08/2021	B	4735	14:20	14:45	13 CALLE B 5-418	OSCAR ESQUITE	MARIO CASTELLANOS	HELMER REYES	1157						
104	05/08/2021	B	4736	15:00	15:35	COLONIA GONZALEZ 3 LOTE 23	JUAN ORELLANA	VICTOR CASTELLANOS	HELMER REYES	1358	0					
105	05/08/2021	B	4737	15:25	16:00	MAYA TEXIL	ISAIAS AROCHE	CARLOS REYES	HELMER REYES	1157						
106	05/08/2021	B	4738	15:45	16:10	KILOMETRO 31 RUTA AL PACIFICO	JUAN ORELLANA	VICTOR CASTELLANOS	HELMER REYES	1358						
							ISAIAS AROCHE	CARLOS REYES	HELMER REYES	1015		0				
										1157						

Fuente: Benemérito Cuerpo Voluntario de Bomberos de Guatemala (2021). *Cabina de la XXIX compañía de Bomberos Voluntarios.*

1.2.2. Registro de unidades de rescate

Las unidades de rescate son los vehículos que se utilizan para realizar las labores de servicio a la comunidad, ya sea atención de emergencias o servicios varios a la comunidad. El único registro que consta de cada unidad es la tarjeta de circulación asociada a cada vehículo, no se cuenta con un registro adicional.

1.2.3. Control de Bomberos Voluntarios

Se lleva el control de los bomberos activos en una hoja electrónica, en la cual en la cual se capturan los datos generales del bombero, nombre, edad, dirección, teléfonos de contacto, clase, rango, entre otros.

1.2.4. Control de equipo de rescate

El equipo de rescate disponible en la compañía se encuentra registrado en cuadernos de papel, el cual incluye notas escritas a mano con la descripción de los cascos disponibles, equipo de oxígeno, herramientas hidráulicas, entre otros.

1.3. Procesos internos

Cada compañía de bomberos alrededor del país lleva sus propios controles y procesos, esto debido a la poca (o nula en algunos casos) automatización tecnológica con la que cuentan, algunos procesos son genéricos y se llevan de la misma forma independientemente del lugar donde este la sede bomberil, sin embargo, los procesos descritos a continuación son los que actualmente se llevan a cabo en el día a día de la vigésimo novena compañía de Amatitlán.

1.3.1. Recepción de llamada de emergencia

El telefonista de turno atienda la llamada entrante en alguna de las líneas telefónicas disponibles, inicia con el llenado de la boleta de solicitud de servicio anotando primeramente el nombre de quien reporta la emergencia, posteriormente pregunta cuál es la emergencia, el telefonista la tipifica y la anota

en la boleta, procede a preguntar la dirección exacta del siniestro o emergencia, pregunta por una ubicación de referencia. El telefonista procede a informar a la persona que se procederá a coordinar el envío del personal disponible con una unidad. Al finalizar la llamada, el telefonista procede a verificar en su hoja de registros que unidad está disponible para atender la emergencia, la destaca al lugar y anota los bomberos que se van dentro de la unidad. Anota la fecha y hora de salida.

Existen casos en los que no hay unidades disponibles en la estación al momento de recibir las llamadas de emergencia, en ese caso se coordina vía radio comunicación para que la unidad que este próxima a finalizar un servicio pueda ir a atender la siguiente emergencia sin necesidad de regresar a la estación (base).

1.3.2. Asignación de insumos a botiquines

Cada unidad de rescate tiene un botiquín físico, el cual debe contener, en la medida de lo posible, los insumos necesarios para atender lesiones comunes, este puede constar de gazas, antibióticos, antinflamatorios, analgésicos, soluciones para desinfectar heridas, tablillas inmovilizadoras, entre otros. Estos botiquines son responsabilidad de los bomberos de turno, ellos serán los responsables de indicar cuando un botiquín este con falta de insumos, así como también deberán informar los insumos utilizados en un servicio en específico.

1.3.3. Asignación de equipo de protección

A cada bombero se le asignara, siempre y cuando haya disponibilidad, un equipo de protección, el cual consta de un casco protector, una casaca de protección, botas, así como el uniforme que los distingue como elemento activo

del cuerpo voluntario de bomberos. Cabe aclarar, que este equipo siempre permanecerá dentro de la compañía, ya que podrá ser utilizado por la siguiente cuadrilla de turno.

2. HERRAMIENTAS TECNOLÓGICAS

A continuación, vamos a conocer las herramientas tecnológicas de las cuales vamos a disponer para darle solución a la problemática antes expuesta. Estas herramientas no son las únicas existentes, hemos seleccionado estas porque son las que se adaptan mejor en este momento para poder iniciar con la automatización de la compañía de bomberos, además de ser de licencia libre, es decir de uso gratuito. Un punto importante para tomar en cuenta es que estas herramientas no requieren de un ambiente con grandes prestaciones para poder ser publicadas en el internet, la solución que se desarrolle con las herramientas siguientes podrá ser desplegada en un servicio de alojamiento web básico, que es uno de los objetivos de este trabajo de tesis.

2.1. Aplicaciones web

El internet se ha convertido hoy en día en una herramienta indispensable en cada hogar, oficina, departamento y compañía, ya que es la principal fuente de conexión entre dispositivos y personas. Pero por sí solo no es el internet el que realiza esta interacción para llevar la información a donde debe llegar, si no son los programas que funcionan ejecutándose en un dispositivo móvil o en una computadora, utilizando la conexión a internet disponible para poder cargar la información que desplegaran al usuario, guardar los datos que se requieran, procesar información o simplemente servir de canal de interacción con usuarios en otras ubicaciones.

En cuanto a estos programas que hacen uso de las conexiones a internet, Gibb, R. (2016) explica que son las denominadas aplicaciones web. Estas

aplicaciones utilizan el internet para poder realizar las tareas para las que fueron creadas, esto conlleva a que no se requiere de una infraestructura local para que puedan ejecutar dichas tareas, únicamente utilizaran el internet para conectarse a los servidores en los cuales consultaran, procesara y guardaran información.

2.1.1. Inicios

En cuanto al inicio de estas aplicaciones, Mateu, C. (2004) explica que en los años 90´s , en un principio, el concepto como tal de aplicación web no existía o no estaba definido como lo conocemos hoy en día, las páginas en ese momento, eran únicamente un conjunto de archivos con contenido HTML, todos estos contenidos eran estáticos, es decir, no tenían interacción alguna con sus visitantes, era requerido que el programador de dicho sitio hiciera cambios manuales a este código para poder cambiar el contenido de estas páginas publicadas.

Fue en 1995 cuando el programador Rasmus Lerdorf puso a disposición al público su lenguaje PHP, el cual era capaz de ejecutar rutinas de código fuente dentro de la paginas estáticas, con esto el contenido de las páginas dejo de ser estático en su totalidad, es aquí con la inclusión de este lenguaje de programación cuando despego el desarrollo de las aplicaciones web. Actualmente muchos sitios tienen como base este lenguaje, el cual ha estado en constante actualización desde esa fecha, hoy en día (octubre de 2021) la última versión estable es la 8.0.11.

En ese mismo año, el navegador web NetScape, el más popular en ese momento, anunciaba una nueva tecnología JavaScript, dicha tecnología permitía a los programadores cambiar el contenido de página web de forma dinámica, lo

cual era algo revolucionario, pues hasta ese momento las páginas web únicamente tenían texto estático, que servía de información a sus visitantes.

En 1996 se lanzó la aplicación HotMail, la cual originalmente no era un desarrollo de la empresa Microsoft, fueron los programadores Sabeer Bhatia y Jack Smith quienes lograron realizar una aplicación capaz de enviar correos electrónicos a sus usuarios registrados en línea, lo cual fue algo revolucionario para el entonces aún incipiente mundo del internet y sus aplicaciones.

En 1997 una nueva tecnología hacia su entrada en escena, Shokwave Flash, la cual servía para agregar contenido interactivo a las páginas web, acciones predeterminadas para controles de usuarios, animaciones en punteros, cuadros de dialogo, mensajes de aleta, entre otros. Esta tecnología se convertiría en ese entonces a la plataforma por excelencia para desarrollar aplicaciones web interactivas.

A partir de aquí, las aplicaciones web se fueron potenciando con tecnologías incipientes y cambiantes, hasta llegar a como las conocemos actualmente, sistemas completos capaces de procesar grandes cantidades de información, servir esta información a miles de personas alrededor del mundo de una manera casi instantánea y hacer mover las grandes operaciones de gigantes industrias globales.

2.1.2. Tipos de aplicaciones web

Existen diferentes tipos de aplicaciones web, estas estarán clasificadas según su propósito, es decir, según las operaciones que realice en su contexto. A continuación, se detallan los diferentes tipos y las diferencias entre cada uno.

2.1.2.1. Aplicación web estática

Aplicación con el fin de no ofrecer contenido interactivo, está pensada únicamente para mostrar contenido estático, el cual no va a cambiar con frecuencia, es ideal para sitios informativos, cuyo contenido será el mismo en un período largo. Por ejemplo, una aplicación que muestre la información sobre las políticas de una empresa, el currículo digital de un profesional o consulta de leyes, todo este contenido no requerirá cambios instantáneos en su estructura o información.

2.1.2.2. Aplicación web dinámica

Aplicación que, según las interacciones o privilegios del usuario, puede cambiar de manera inmediata su contenido. Estas aplicaciones regularmente constan de varios servicios internos que son los encargados de proveer la información en tiempo real para mostrar. Usualmente se conectan a una base de datos de donde obtendrán el contenido solicitado por el usuario. Estos son sistemas más complejos, su código fuente es más amplio, se requiere de requisitos específicos para poder montar o publicar la página al algún servidor. Estas aplicaciones son las más comunes hoy en día, por ejemplo, la aplicación de un banco, la cual provee del saldo y operaciones recientes sobre una cuenta bancaria. Este contenido estará siempre en contante cambio a medida que las operaciones sobre la cuenta sean más frecuentes.

2.1.2.3. Aplicación web de comercio electrónico

Este tipo de aplicación se basa en simular una tienda de artículos física, en donde podremos encontrar secciones de artículos de un tópico especifico, un carrito de compras en el cual ir depositando los artículos que hayamos marcado

para compra. Finalmente tendremos un espacio que simulará la caja de cobro, el cual por lo regular tendrá el nombre de *checkout,* en donde indicaremos nuestro método de pago y la dirección de entrega de los artículos comprados.

Un ejemplo de este tipo de aplicaciones web podríamos mencionar amazon.com, en donde podremos ver claramente la sección de compras, pago, detalle de los artículos, entre otros.

2.1.2.4. Portal web app

Un portal web *app* se refiere a que la aplicación cuenta con una página principal, la cual permite el acceso a diferentes sistemas dentro de la aplicación. Por ejemplo, en la página principal podremos seleccionar si queremos ir a la sección de correo electrónico, a la sección de foros, sección de noticias, entre otros.

Un ejemplo de un portal web *app* es Google.com en el cual podemos seleccionar en el menú de aplicaciones a donde queremos ir, correo electrónico (Gmail), sección de videos y *streaming* (YouTube), entre otros.

2.1.2.5. Gestor de contenidos

Este tipo de aplicación web, como su nombre lo indica, se encarga de gestionar el contenido que se presenta a los usuarios de la aplicación, actualiza las secciones de contenido completas, para esto cuenta con Content Management System (CMS), el cual es un sistema que se encarga de administrar el contenido que se sirve en la página, administra las contribuciones al contenido por parte de los usuarios autorizados. Uno de los gestores de contenidos más

populares es *WordPress,* el cual es un sistema que permite crear, mantener y administrar contenido que será publicado en una página o aplicación web.

2.1.3. *Front end*

Toda aplicación web requiere de un diseño, este diseño obedece a los propósitos para los que fue concebida la aplicación. Dentro de este diseño se encuentran dos grandes secciones o apartados en los cuales podemos dividir una aplicación, el *front end* y el *back end*, ambos con tareas y propósitos bien definidos. A continuación, vamos a profundizar en ambos conceptos.

En cuanto al *front end,* es todo lo que la aplicación muestra al usuario, es decir, la cara de la aplicación, lo que los usuarios verán al momento de cargar o ejecutar la aplicación, los colores, botones, estilos, imágenes, cuadros de texto, secciones, tablas, entre otros. (Bautista García, I. 2021)

En palabras más técnicas, es todo el código que se ejecuta del lado del navegador web del usuario, se le puedo denominar también como aplicación cliente. Esto quiere decir que el *front end* no está libre de código, por el contrario, aquí hay mucho código que se ejecutara para poder formas las vistas de la aplicación de cara al mundo, al usuario que es quien va a interactuar con la aplicación. Como mencionamos, este código se ejecutará en la computadora del usuario, localmente, será el navegador web el encargado de interpretar y ejecutar lo que el código contenido en el *front end.*

Del lado del *front end* se encuentran lenguajes de programación que se ejecutan del lado del cliente, es decir, dentro de la página que se está presentando, estos lenguajes (por mencionar algunos) son:

- HTML

Lenguaje de marcas de hipertexto (por sus siglas en inglés), en cuanto a este lenguaje, Cobo, Angel (2005) explica que es un lenguaje interpretado por el navegador web para mostrar los distintos componentes de la página o aplicación web.

- CSS

Hoja de estilos en cascada (por sus siglas en inglés), en cuanto a este lenguaje, Cobo A. (2005) explica que lenguaje utilizado para dar formato y estilos definidos a todos los elementos HTML que contenga la aplicación.

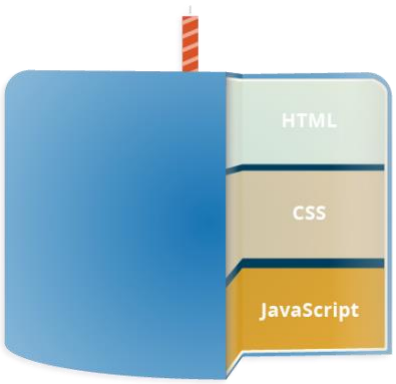
- JavaScript

En cuanto JavaScript, es un lenguaje de programación interpretado del lado del cliente, es decir interpretado por el navegador web, es el encargado de darle interacción a la aplicación, manejando eventos que desencadenaran acciones definidas por porciones de código contenido dentro de secciones especiales dentro del código HTML. Aporta la parte dinámica a la página o aplicación web, gracias a este la aplicación podrá ejecutar funciones complejas dentro de la página. (Cobo A, 2005)

Podemos ver como interactúan estas tecnologías si hacemos una analogía de un pastel compuesto por capas, cada una de estas capas es una tecnología, mientras más adentro vamos en el pastel, nos encontramos con una tecnología, por ejemplo, en la capa más profunda esta JavaScript la cual contiene todo el trabajo pesado, las secuencias de control complejas, en la capa de en medio vemos a CSS, la cual dará todo el formato de como ver los componentes

a mostrar, finalmente en la primer capa tenemos a HTML, el cual mostrara al usuario todo el contenido procesado en las capas anteriores de una forma amigable legible, con el formato correcto y con el resultado de la ejecución del código.

Figura 2. **Capas de tecnologías web estándar**



Fuente: Mozilla (2021). *Una definición de alto nivel*. Consultado el 20 de septiembre de 2021. Recuperado de https://developer.mozilla.org/es/docs/Learn/JavaScript/First_steps/What_is_JavaScript.

2.1.4. Entorno de desarrollo integrado

Un entorno de desarrollo integrado (IDE por sus siglas en inglés) es un *software* que se encarga de mostrar en una interfaz gráfica de usuario (GUI por sus siglas en inglés) un entorno amigable en donde están todas las herramientas disponibles de un lenguaje de programación. Un IDE típicamente está compuesto por los siguientes componentes principales:

- Editor de Código Fuente

Sera el encargado de analizar el código fuente, es decir, el texto que se ingresa en el área designada para este propósito, entre las funciones principales podemos mencionar el resaltar errores de sintaxis, proveer de manera visual los diferentes contextos del código, indicar elementos pertenecientes a diferentes clases y en algunos casos proveer inteligencia de código el cual nos dará sugerencias para poder completar porciones de código y de esta manera hacer más rápido y eficiente la generación del mismo.

- Automatización de Compilaciones

Se encargará de realizar la traducción de todo el código fuente escrito a un lenguaje de máquina, un lenguaje de bajo nivel, el cual es compresible para la computadora. En este punto también realizara una primera comprobación de tipos, análisis de sintaxis, análisis de semántica en algunos caos.

- Depurador

Se utiliza para ejecutar el resultado del código fuente compilado, este se ejecutará paso por paso, cada paso estará definido por el programador, nos dará información de esta ejecución paso por paso con el fin de poder detectar comportamientos anómalos, errores en tiempo de ejecución, entre otros.

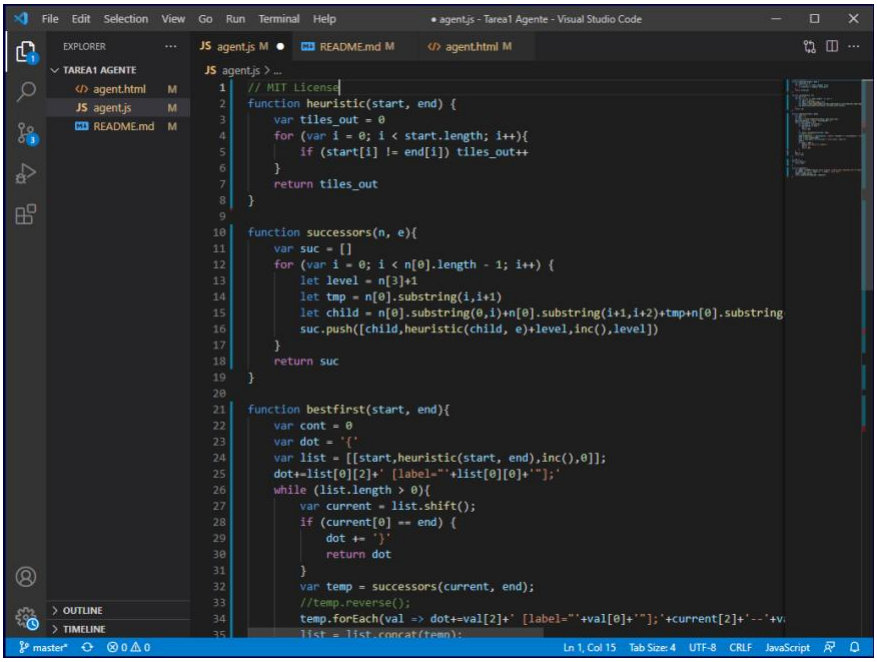
2.1.4.1. Visual Studio Code

El entorno de desarrollo integrado que vamos a utilizar para desarrollar el *software* para la compañía de bomberos será Visual Studio Code. Este IDE fue desarrollado por Microsoft y cuenta con una licencia gratuita para poder utilizarlo, además de proveer un entorno completo, cumple con las tres características mencionadas en la sección anterior. Cuenta con soporte multilenguaje, que, para

nuestros usos, vamos a utilizarlo para poder generar, editar y compilar código fuente en lenguaje PHP y JavaScript.

Este es un IDE relativamente nuevo, ya que fue en el año 2015 cuando vio la luz, bajo la licencia MIT la cual permite poder utilizarlo de forma gratuita, siempre y cuando se respete el acuerdo de licencia que se debe aceptar cuando instalamos el *software*. Una de las características de este IDE que nos serán muy útiles es la capacidad de poder instalar *Plugins* los cuales son pequeñas características extra que nos proporcionaran soporte para algunas funcionalidades dentro del sistemas a desarrollar (en los capítulos siguientes vamos a ver estar características).

Figura 3. **Apariencia de Visual Studio Code**



Fuente: elaboración propia, realizado con Visual Studio Code.

2.2. Restful API

Un API (Interfaz de programación de aplicaciones por sus siglas en inglés) es un conjunto de procedimientos o funciones complejas para poder conectar dos sistemas. Podemos decir, utilizando la analogía de un contrato de servicios, que un API es como un contrato entre un proveedor de información y un consumidor de esta información, en este contrato se establecerá el contenido que se necesita por parte del consumidor (cuando se hace la llamada al servicio) y lo que el productor enviara (respuesta del servicio). En otras palabras, un API le permitirá administrar datos que se encuentran en una capa diferente del sistema que se está utilizando.

2.2.1. Rest

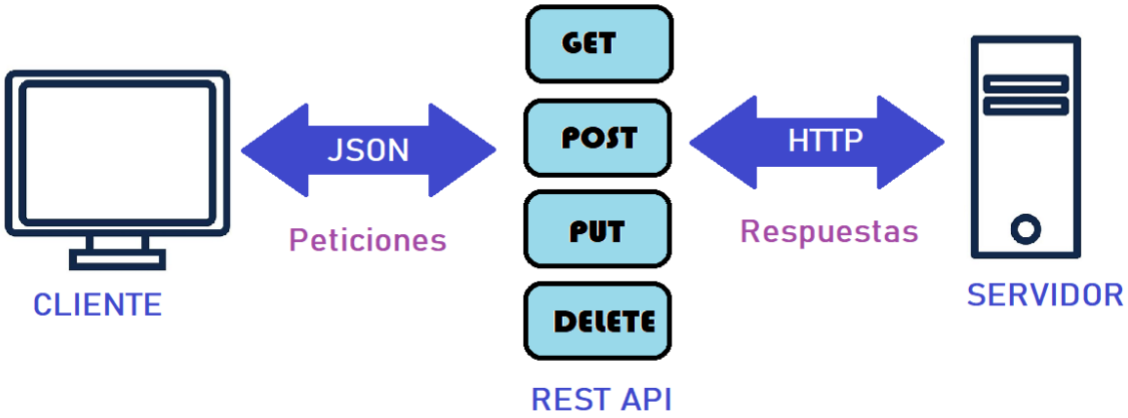
Representational state transfer, en cuanto a este concepto, Wikipedia (2021) explica que es una interfaz que utilizamos para poder conectar dos o más sistemas por medio del protocolo HTTP. Por medio de esta interfaz vamos a poder obtener y generar datos como resultado de ejecutar las operaciones contenidas dentro de esta. Las respuestas y peticiones a esta interfaz serán en formatos específicos conocidos, tal es el caso de los dos más populares JSON y XML.

Tabla I. **Métodos de petición HTTP**

Verbo	Función
Get	Obtener datos (leer/listar)
Post	Enviar datos (guardar o generar nuevo)
Put	Enviar datos (actualizar)
Delete	Elimina un dato específico
Options	Describe las opciones de comunicación del recurso
Patch	Aplica modificaciones parciales al recurso destino
Head	Realiza una petición como un get pero sin incluir el cuerpo de la respuesta, únicamente el encabezado
Trace	Realiza una prueba de bucle de retorno a lo largo de la ruta del recurso destino

Fuente: Broadcom (2013). *Verbos de HTTP.* Consultado el 26 de septiembre de 2021. Recuperado de https://ftpdocs.broadcom.com/cadocs/0/CA%20AppLogic%203%207-ESP/Bookshelf_Files/HTML/AppLogicDoc/1937891.html.

Figura 4. **Esquema de trabajo API rest**



Fuente: elaboración propia, realizado con MsPaint.

2.2.2. Restful

En la sección anterior vimos como definir un API REST, siguiendo con este concepto vamos a ampliar el espectro para completar el termino RESTful , para poder diferenciar REST de RESTful debemos entender que, si bien es cierto en algunas referencias o definiciones nos indican que son sinónimos esto no es del todo cierto, ya que podemos definir a REST como la arquitectura de un API basado en los protocoles HTTP y RESTful como los servicios contenidos dentro del API los cuales son programas de *software* basados en la arquitectura REST.

En cuanto a una RESTFul API*,* redhat (2020) explica que esta implementación está concebida para que cualquier aplicación de *software* pueda Consumir cada uno de los recursos que el API pone a disposición, indistintamente del lenguaje en el que este desarrollada la aplicación que realiza las peticiones o la plataforma desde donde se realicen estas peticiones, ya que las respuestas serán estandarizadas en los formatos mencionados anteriormente (JSON, XML).

Un componente fundamental en un RESTful API es el HTTP *Status Code*, que es quien representa el código de estatus de la petición enviada, para así por su parte la aplicación cliente sepa qué hacer con la respuesta obtenida, por ejemplo, si esta tuvo éxito o fue denegada.

Tabla II. **Códigos de estado HTTP**

Código	Significado
200	OK (resultado de éxito)
201	Creado
304	No modificado
400	Petición errónea
401	No autorizado
403	Prohibido
404	No encontrado
422	Entidad no procesable
500	Error interno del servidor

Fuente: Siteground (2015). *Códigos de estado HTTP explicados.* Consultado el 29 de septiembre de 2021. Recuperado de https://www.siteground.es/kb/codigos-error-http-explicados.

2.2.3. *Back end*

Una de las capas fundamentales en la arquitectura de una aplicación web es el denominado *back end* el cual consiste en toda la lógica de negocio que está detrás de la capa de presentación de datos al cliente o usuario, esta primera capa la discutimos en el capítulo anterior, el denominado *front end*.

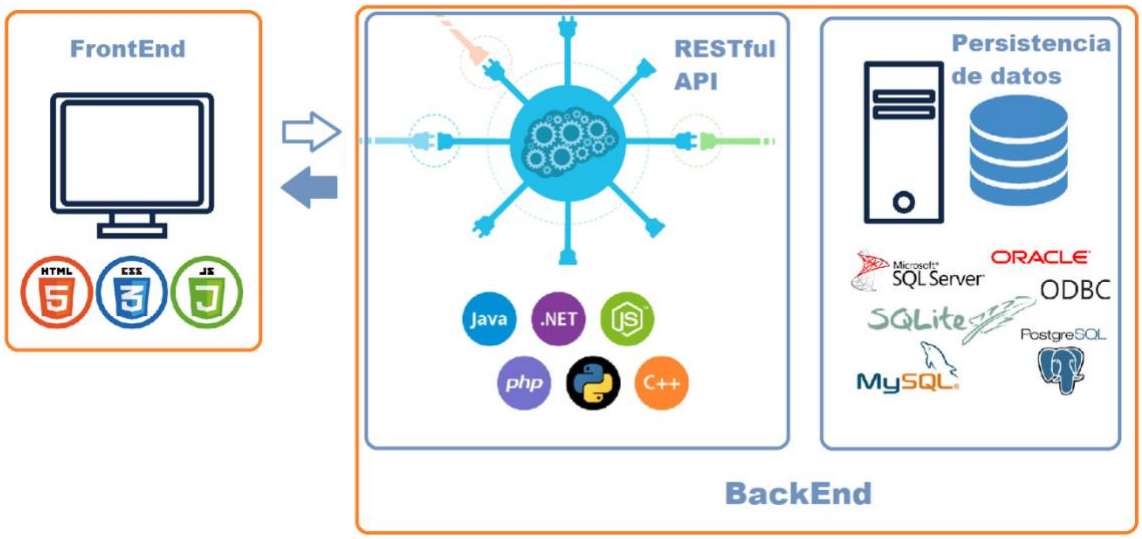
En cuanto al *back end,* es quien trabaja tras bambalinas para poder proveer de datos al *front end*, para poder proveer de estos datos, se valdrá de servicios, los cuales están definidos y programados dentro de esta aplicación. Estos servicios serán los encargados de conectarse a otras fuentes de datos, como por ejemplo bases de datos locales o remotas, procesar estos datos y enviarlos como respuesta a las peticiones realizadas desde los clientes. (Bautista García, I. 2021)

En la sección anterior pudimos ver el concepto de un RESTful API, pues bien, el *back end* de una aplicación web por lo general está compuesta por una API de este tipo, la cual ofrecerá puntos de conexión para poder atender las diferentes peticiones de solicitud de datos, creación de nuevos datos, edición de datos, eliminación de datos, entre otros. Cabe resaltar que cada uno de estos puntos de conexión serán los encargados de proveer la lógica de negocio con la que funcionara nuestra aplicación.

Esta capa de la aplicación web, no es visible para el usuario, únicamente interactúa a nivel de comunicación HTTP con el *front end*, tampoco contiene ningún elemento gráfico, es decir, es una capa abstracción de *software*.

El *back end* tiene la capacidad de conectarse a otras APIs que proveen datos, por ejemplo, servicios meteorológicos, servicios de información en vivo, servicios bancarios, entre otros. Pero no solo se limitan a consumir otros servicios, una de las principales funciones de esta capa es la conexión entre el usuario (*front end)* y la capa de persistencia de datos (base de datos). Esta capa gestionara como debemos solicitar los datos, el filtrado de estos datos, el formato de cómo se deberán presentar, la ofuscación de datos sensibles, cálculo de resultados con base en los datos obtenidos, entre otros.

Figura 5. **Esquema *front end* y *back end***



Fuente: elaboración propia, realizado con MsPaint.

2.3. Bases de datos *open source*

Una aplicación web que se compone de un *front end* y *back end* va a intercambiar información entre estos componentes para presentarla al usuario, esta información surge de una fuente de datos, la cual provee a la capa intermedia *back end* de estos datos para que puedan ser procesados y enviados al *front end*. Esta fuente de datos es lo que comúnmente conocemos como una base de datos, la cual es la encargada de administrar los datos generados a través de las interacciones del usuario con el *front end*.

En cuanto a una base de datos *open source* o de código abierto, Ankush (2021) explica que es todo aquel sistema administrador de base de datos cuyo motor o código base es de uso libre. Según los términos de este tipo de licencia

este *software* lo podremos usar, distribuir, descargar y modificar el código fuente para aspectos de mejora.

Las bases de datos, en general, se dividen en dos grandes grupos, los cuales podemos categorizar de la siguiente manera:

- Base de datos relacional

Este es el enfoque tradicional de una base de datos, almacena los datos respetando el modelo relacional y las reglas de normalización, se compone de tablas, estas a su vez contienen filas y columnas que en conjunto forman los datos. Estas tablas contienen relaciones entre sí para poder establecer principios como el de integridad referencial, consistencia de datos, datos únicos, entre otros. Su lenguaje principal de consulta es el Sql.

- Base de datos No Relacional

También llamadas NoSql, se caracterizan por no pertenecer al modelo relaciona, el lenguaje Sql no es su principal lenguaje de consulta, aunque si lo pueden soportar, los datos no tienen que estar relacionados entre sí, podemos enfocar estas bases de datos a colecciones completas de datos, a almacenar documentos. Su arquitectura es distribuida, por tal razón son más escalables horizontalmente lo cual conlleva a una mejor tolerancia a fallos. Diseñadas para poder analizar y soportar grandes volúmenes de datos.

2.3.1. Ventajas

Una base de datos *open source* frente a una que no lo es, es decir, una base de datos de licencia privada tiene algunas ventajas, que deberemos evaluar

según la implementación que queramos hacer. La primer gran ventaja, pero no la única, es principalmente el tema económico ya que nos ahorramos el costo de la licencia, sin embargo, el soporte que queramos a partir del uso del producto si debemos pagarlo, ya que el tema de la distribución y uso es lo único que cubre la licencia gratuita, el soporte al producto es un tema aparte.

A continuación, vamos a listar algunas diferencias que podemos tomar como ventajas, según sea el caso e intereses, de una base de datos *open source* frente a una que no lo es.

- Podemos utilizar el producto gratuitamente.
- Libre para modificar el código fuente base.
- No hay cargo adicional por implementar varias instancias.
- Variedad de sistemas libres.
- Poder utilizar optimizaciones o extensiones de compatibilidad por terceros que lo publiquen.
- Contar con soluciones para pequeñas empresas, no es necesario implementar un sistema completo para algo que está iniciando o no tendrá una transaccionalidad pesada.
- Implementaciones en Hostings compartidos para poder utilizarla en la publicación de una aplicación web.
- Compatibilidad con sistemas operativos *open source* y acceso a compilaciones específicas.

2.3.2. Aplicaciones en la actualidad

Las bases de datos *open source* en la actualidad han tomado un mayor auge, ya que sus características y bondades frente a las de pago las hacen una solución ideal en muchos casos, ya que para iniciar con un proyecto que no

requiera de un potencial alto de inicio, una solución *open source* queda a la medida.

Existen muchas soluciones en la actualidad en cuanto a bases de datos *open source* se refiere, vamos a mencionar las que sobresalen hoy en día.

Entre las opciones de bases de datos relaciones tenemos:

- MySql (la opción más popular)
- PostgreSql
- MariaDB (derivado de MySql, optimizado para grandes cargas de datos)

Del lado de la NoSql podemos mencionar:

- MongoDB (la opción más popular)
- CouchDB
- Cassandra
- Neo4j
- Redis

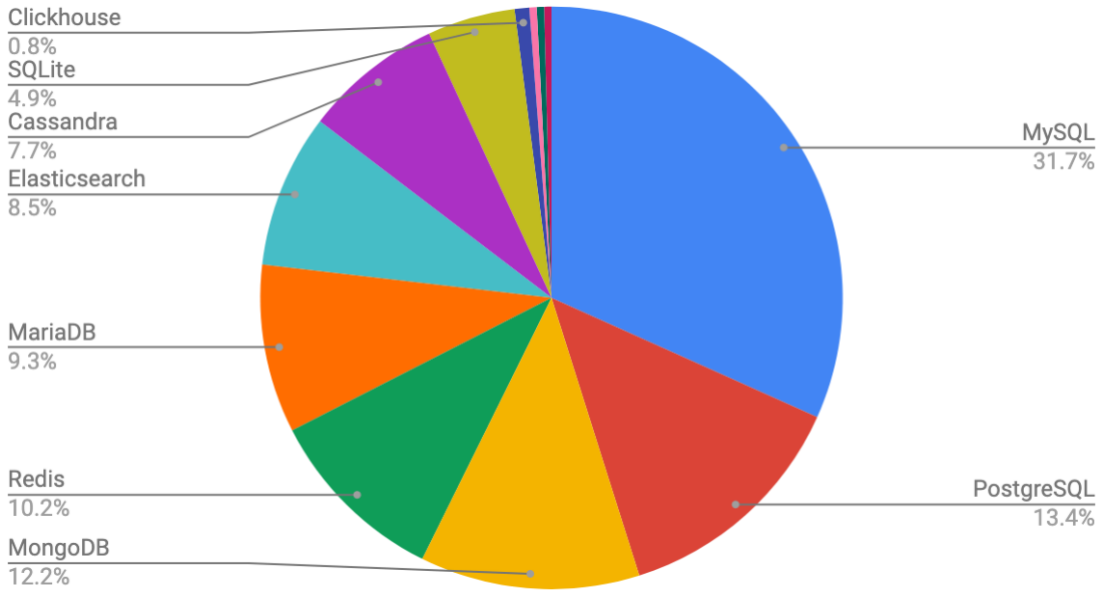
Actualmente el uso de cualquiera de las bases de datos antes mencionadas ira en función de lo que estemos buscando resolver, por ejemplo, si queremos analizar grandes cantidades de información, procesar datos dejando de lado un poco la relación nativa, debemos entonces utilizar en *software* de base de datos NoSql (no relacional) ya que la implementación de estos sistemas está dirigida para estos propósitos.

En caso contrario, si nuestra necesidad es mantener la relación entre datos, integridad referencial, valores únicos, atomicidad, entre otros. La solución

será la implementación de un *software* de base de datos relacional, el sistema tradicional.

MySql y MongoDB han tomado una mayor popularidad en la comunidad de desarrollo, en el ambiente relacional y NoSql respectivamente. Para el año 2019 y 2020 ambas bases de datos se repartían el 31.7 % y 12.2 % del uso en general.

Figura 6. **Popularidad de bases de datos *open source***

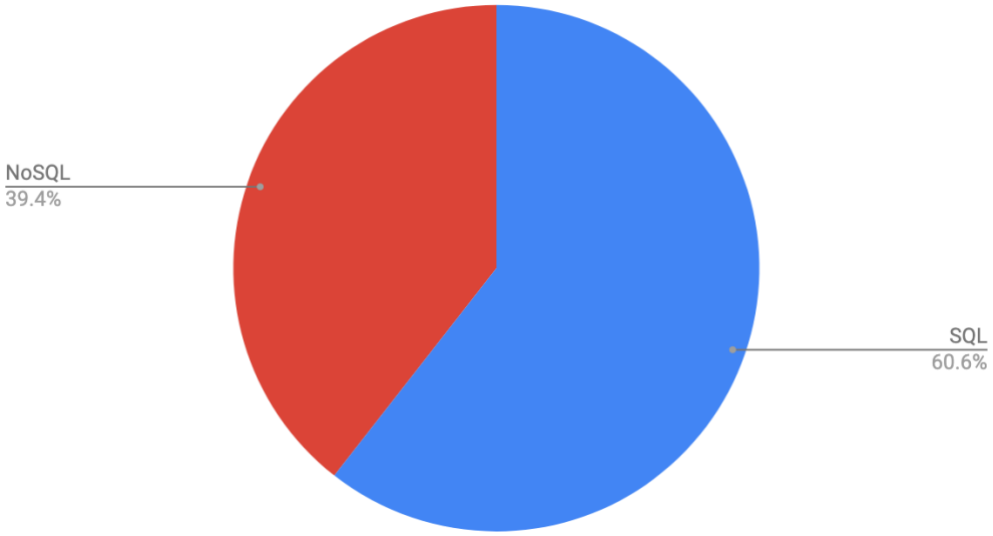


Fuente: Dzone (2019). *Open Source Database Report.* Consultado el 26 de septiembre de 2021. Recuperado de https://dzone.com/articles/2019-open-source-database-report-top-databases-pub.

Claramente se evidencia quienes lideran el mercado, el menos en nivel de popularidad entre la comunidad de desarrollo, esto se debe en primer lugar a que como bien hemos resaltado anteriormente proveen despliegues sencillos y un rendimiento capaz de satisfacer la etapa inicial e intermedia de un proyecto. Por

su lado, MySql es la base de datos por excelencia en los web hostings, ya que, por ser un servicio compartido, la instancia principal de estos servicios cuenta con este motor de base datos instalado, de este motor se otorga un segmento para cada espacio de hosting adquirido, por esta razón una gran parte de los sitios de pequeñas y medianas empresas que deciden utilizar una solución compartida para desplegar sus aplicaciones utilizan MySql casi como un estándar de facto.

Figura 7. **Popularidad SQL vs NoSql *open source***



Fuente: Dzone (2019). *Open Source Database Report.* Consultado el 26 de septiembre de 2021. Recuperado de https://dzone.com/articles/2019-open-source-database-report-top-databases-pub.

En los siguientes capítulos vamos a proponer el diseño de una aplicación web que, entre otras cosas, utilizara un *software* relacional de bases de datos para dar solución a la problemática expuesta en el capítulo 1 de este trabajo de tesis.

3. APLICACIÓN QUE SE DESARROLLARÁ

En los capítulos anteriores hicimos un repaso por las tecnologías *open source* que hemos seleccionado para poder dar una solución viable a la problemática actual de la vigésimo novena compañía de Bomberos Voluntarios, cada una de estas han sido analizadas para poder ver la factibilidad de cada una durante el proceso de desarrollo.

A continuación, vamos a detallar dicha propuesta, daremos a conocer sus principales características de diseño, argumentos técnicos y la forma en que será desplegada para la disponibilidad de la compañía de bomberos en mención.

3.1. Propuesta para solucionar la problemática actual

Se desarrollará una aplicación web del tipo portal web *app*, ya que constara de una página inicial en la cual se mostrará información al público en general, además contara con un apartado para poderse dirigir al sistema de gestión de servicios y emergencias, por medio de una autenticación por usuario y contraseña.

Contará también con las siguientes secciones:

- Portada
- Acerca de nosotros
- Contáctenos
- Inicio de sesión

- Menú catálogos
  - Catálogo de bomberos activos
  - Catálogo de unidades motorizadas
  - Catálogo de herramientas y equipo de rescate
  - Catálogo de insumos
  - Catálogo de servicios/emergencias
- Menú operaciones
  - Registro de llamadas
  - Asignación de unidades motorizadas
  - Menú reportes
    - ✓ Reporte de emergencias
    - ✓ Reporte de servicios

Cada una de las secciones anteriores contara con una página web, la cual será desarrollada de manera intuitiva para que los usuarios puedan comunicarse e informarse de una manera adecuada acerca de la compañía de bomberos. Cada una de estas secciones de detallará en el siguiente capítulo.

El *front end* se desarrollará en el *framework* Angular, el cual es un *framework open dource* desarrollado por *Google*, este marco de trabajo se ha convertido en uno de los más populares hoy en día para el desarrollo de páginas web, el lenguaje que se utiliza es *TypeScript* el cual tiene como base JavaScript.

Del lado del *back end* vamos a desarrollar un RESTFul API utilizando el lenguaje PHP, escogimos este lenguaje por dos razones muy importantes, la primera es que es un lenguaje *open source* que se adapta y especializa al desarrollo web. En segundo lugar, porque, como detallaremos más adelante, el sitio donde será desplegada la solución final provee un ambiente de ejecución para APIs PHP.

Para la capa de persistencia de datos hemos escogido de igual forma a conveniencia de nuestra arquitectura el motor de base datos relacional MySql, ya que, por la misma razón del API, el sitio donde realizaron el despliegue de dicha aplicación nos provee de una instancia de MySql, lo cual nos facilitara el tema de compatibilidad entre plataformas y aplicaciones.

3.2. Arquitectura

Como mencionamos en la propuesta de la aplicación, cada una de las áreas de la aplicación se desarrollará bajo un esquema web, utilizando los lenguajes y marcos de trabajo más populares y factibles que podemos encontrar hoy en día.

- *Front end*: desarrollado en Angular, con librerías de Angular Material.
- *Back end*: RESTFul API desarrollada con PHP.
- Base de Datos: MySql.

La siguiente imagen nos muestra de manera resumida como interactuara cada componente del sistema y que lugar se encuentra de la arquitectura propuesta.

Figura 8. **Arquitectura aplicación web**

Fuente: elaboración propia, realizado con MsPaint.

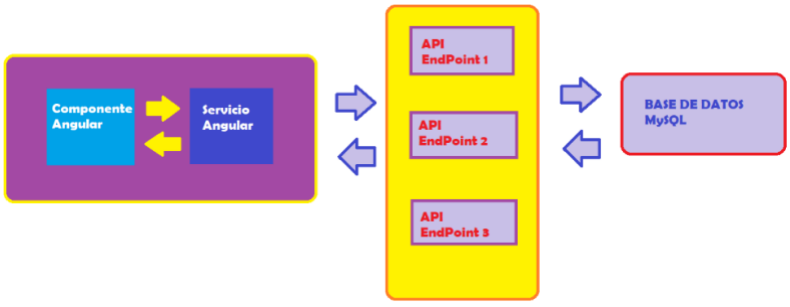
3.2.1. *Front end*

La aplicación web que se encarga de mostrar toda la información del sistemas se desarrollara en el *framework* Angular, dentro de este marco de trabajo vamos a implementar código HTML, CSS, y TypeScript, en este apartado se definirán los componentes que se encargaran de renderizar cada una de las páginas de las que se componen cada sección del sistema, además se definirán servicios para poder consumir todos los *endpoint* que nos provea el API, tanto para obtener datos, guardar datos, editar datos y eliminar datos.

3.2.2. *Back end*

El API será desarrollada en el lenguaje PHP dadas las bondades del lenguaje, su especialización web y la conveniencia para poder desplegar la aplicación en servicios web hostings compartidos. La aplicación tendrá definidas rutas de los diferentes endpoint que se proveerá al consumidor, además se definirán los verbos HTTP que vimos en el capítulo anterior (*get*, *post*, *put*, *delete*, entre otros).

Figura 9. **Arquitectura de componentes**



Fuente: elaboración propia, realizado con MsPaint.

3.3. Diseño de base de datos

A continuación, se vamos a representar en el diagrama las tablas que serán utilizadas para guardar la información del sistema, así como definir las relaciones existentes entre estas tablas y las propiedades de cada una, así como definir las propiedades únicas que servirán para identificar un registro en todo el sistema.

3.4. Despliegue de la aplicación

Para realizar el despliegue de la aplicación (*Deployment*) hemos contratado el servicio de un web hosting, se adquirió el dominio www.cvb29compania.com para asociar este dominio con nuestro sitio web.

Un web hosting no es más que un servicio de alojamiento web, el cual funciona de la siguiente manera, existe una computadora física , un servidor, el cual funciona ininterrumpidamente, salvo en las ocasiones de mantenimientos programados o caída del servicio por causas ajenas al proveedor, este servidor cuenta con un *software* especializado para poder administrar cada sitio web que es alojado en él, cuando adquirimos un servicio de web hosting estamos adquiriendo en renta una porción del espacio de este servidor ininterrumpido, en este espacio vamos a alojar todos nuestros componentes que son necesarios para que nuestra aplicación web pueda estar disponible en todo momento para sus usuarios.

El dominio es la dirección de donde podrá ser ubicada nuestra aplicación web, es decir la referencia que al momento de ser ingresada en los navegadores web, estos sepan hacia dónde dirigirse.

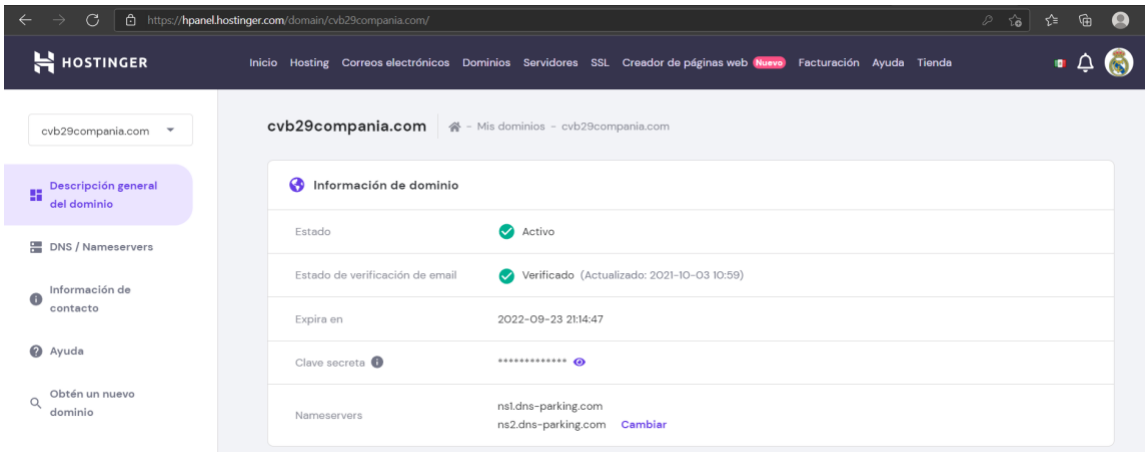
Este servicio de alojamiento también nos dará acceso a una instancia de base de datos MySql por tal razón a conveniencia nuestra indicamos en los capítulos anteriores que la base de datos a utilizar seria la ya mencionada.

Figura 10. **Analogía de alojamiento web**

Fuente: Hostinger (2020). *¿Qué es un hosting y como funciona?*. Consultado el 26 de septiembre de 2021. Recuperado de https://www.hostinger.es/tutoriales/que-es-un-hosting.

Para nuestro caso, hemos seleccionado el proveedor HOSTINGER, ya que sus precios son accesibles, están dentro del rango que se puede permitir la estación de bomberos, además de contar excelentes referencias en calidad de servicio. Como un inicio se adquirió el servicio por 1 año, así como la compra del dominio proprio www.cvb29compania.com también se realizó por 1 año, siendo la fecha de renovación del servicio en septiembre de 2022.

Figura 11. **Hosting y dominio adquirido**



Fuente: Hostinger (2021). *Panel de control de dominio.* Consultado el 26 de septiembre de 2021. Recuperado de https://hpanel.hostinger.com/domain/cvb29compania.com/domain-overview.

En este alojamiento vamos a tener nuestra aplicación web, tanto *front end* como el *back end*, ya que este servicio nos da esa capacidad, vamos a centralizar nuestro sistema en este dominio para que esté disponible las 24 horas y 365 días del año para la operación de la vigésimo novena compañía de Bomberos Voluntarios.

4. ESTRUCTURA DE LA APLICACIÓN

La aplicación que vamos a desarrollar para darle solución a la problemática actual de la vigésimo novena compañía de Bomberos Voluntarios estará distribuida en diferentes secciones y apartados, cada uno con un propósito especifico, a continuación, vamos a describir cada una de estas secciones en las cuales vamos a encontrar las pantallas de catálogos (donde se almacenará la información que alimentará la operación) y las pantallas de operación (atención de emergencias y servicios).

4.1. Pantalla de inicio

La pantalla de inicio será la página que se cargará inicialmente cuando el visitante se dirija al dominio cvb29compania.com, contendrá información general, la cual se detalla a continuación.

4.1.1. Portada

En esta sección vamos a presentar la información que el visitante del sitio vera como primera instancia, la primera información que se cargará en la pantalla, la cual contendrá información sobre las generalidades de la compañía.

Una imagen de la fachada de sus instalaciones, el nombre completo de la compañía, el escudo de la compañía serán los elementos principales de la pantalla de portada.

4.1.2. Acerca de nosotros

En esta sección se mostrará una breve reseña de la compañía de bomberos, su historia, quienes conforman la actual junta de oficiales, la misión y visión de la compañía.

Los elementos principales de esta sección serán, el escudo de la compañía, el cuadro de texto con la reseña e información adicional, una imagen del interior de la compañía en donde se muestra la denominada Sala de Máquinas, dicha sala se refiere al área donde se estacionan las unidades de rescate.

4.1.3. Contáctenos

En esta sección se mostrará la información de contacto de la compañía de bomberos, se mostrarán los números de teléfono disponibles para reportar emergencias, la dirección física de la estación de bomberos y el correo electrónico de contacto, en el cual el público en general podrá enviar sugerencias, comentarios, entre otros.

4.2. Menú catálogos

En este menú se mostrará el listado de los catálogos disponibles en el sistema, estos catálogos captaran toda la información necesaria para poder alimentar a los procesos operativos que se listaran más adelante.

4.2.1. Catálogo de bomberos activos

Este catálogo se encargará de almacenar toda la información referente a los bomberos que forman parte de la compañía. Desde esta pantalla se podrá obtener un listado de todos los bomberos activos en la compañía, sed podrá dar de alta a un nuevo elemento bomberil, se podrá editar la información general de un bombero, así como también se podrá dar de baja a un elemento cuando sea necesario.

Los atributos que se deberán ingresar para dar de alta a un nuevo bombero son los siguientes:

- Nombres
- Apellidos
- Fecha de nacimiento
- Dirección domiciliar
- Teléfono residencial
- Teléfono móvil
- Correo electrónico
- Fecha de alta
- Categoría
- Turno

4.2.1.1. Dar de alta a bombero

Para poder ingresar esta información se proveerá de un formulario, para poder dirigirse a este formulario, en la pantalla inicial del catálogo se deberá presionar sobre el botón Agregar Nuevo, se mostrará automáticamente la página que contiene el formulario para iniciar con el proceso. Cuando se termine de

ingresar la información se deberá presionar el botón GUARDAR para grabar la información en el sistema.

4.2.1.2. Editar un registro de bombero

Si se tiene la necesidad de corregir uno o varios datos específicos de un registro de un bombero en el catálogo, debemos listar los bomberos existentes presionando el botón Mostrar Datos, para que el sistema muestre el listado de registros de bomberos que se tienen hasta ese momento. En dado caso queramos hacer una búsqueda más específica, podemos hacer uso de los filtros que nos provee la pantalla. Cuando se ubique en la pantalla el registro del bombero que deseamos editar, bastara con presionar el botón identificado con el nombre Editar, esta acción dirigirá al usuario al formulario de altas de registros con la información precargada del bombero, es aquí donde debemos actualizar el o los campos que necesitemos, al finalizar la edición debemos presionar el botón Guardar para que los cambios sean grabados en el sistema.

4.2.1.3. Dar de baja a un bombero

Cuando es necesario notificar al sistema de la baja de un bombero, se deberá dirigirse nuevamente al listado de bomberos activos, presionar el botón con el nombre Dar de baja, se presentará al usuario una pantalla emergente, en el cual se preguntará si realmente se le quiere dar de baja al bombero, en caso de presionar que sí, se pedirá que se ingrese un motivo de la baja. Para terminar con el proceso de baja se deberá presionar el botón Guardar para grabar la baja del bombero en el sistema.

4.2.2. Catálogo de unidades motorizadas

En este catálogo vamos a almacenar las unidades de rescate con las que cuenta la compañía de bomberos. Estas unidades de rescate tendrán los siguientes atributos:

- Tipo de vehículo
  - Ambulancia
  - Panel acondicionada
  - Pickup
  - Motobomba
  - Sisterna
  - Camioneta
  - Sedan
  - *Hatch back*
  - Motocicleta
- Marca
- Línea
- Modelo
- Número de ejes
- Origen
  - Donación
  - Compra
  - Préstamo
- Estatus
  - Disponible
  - En reparación
  - De baja
  - En servicio

- Fecha de ingreso
- Fecha de baja
- Razón de baja
- Observaciones
- Número de unidad
- Código de unidad

La pantalla proveerá los controles para poder ingresar, editar y dar de baja a una unidad de rescate.

4.2.2.1. Agregar unidad motorizada

Se deberá dirigir al menú de unidades motorizadas, presionar el botón Agregar Nuevo, se presentará en pantalla el formulario de captación de datos de la unidad motorizada, al terminar de llenar los datos requeridos se deberá presionar el botón Guardar para que se graben los datos en el sistema. El usuario automáticamente será redirigido al listado de unidades motorizadas existentes donde se podrá ver la unidad recientemente ingresada.

4.2.2.2. Cambiar estatus unidad motorizada

Se proveerá de una acción directa desde el catálogo de unidades motorizadas para poder cambiar directamente el estatus asignado a una unidad. Para este propósito se deberá presionar el botón de Cambiar Estatus que se encontrará en las propiedades de cada registro del listado de unidades, se deberá seleccionar el estatus a asignar y presionar el botón Guardar para asignar el nuevo estatus a la unidad motorizada.

4.2.2.3. Editar unidad motorizada

Para poder actualizar o editar la información relacionada a cada unidad motorizada se deberá listar las unidades motorizadas disponibles en la pantalla del catálogo de unidades motorizadas, identificar el registro a editar y presionar sobre las propiedades del registro, debemos entonces seleccionar la opción Editar, automáticamente el sistema mostrará el formulario de registro de unidades con la información precargada de la unidad. En este punto debemos editar la información que se desea. Al momento de terminar de actualizar la información deseada debemos guardar los cambios para terminar con el proceso, para esto bastará con presionar el botón Guardar para que el sistema proceda a grabar la información registrada.

4.2.2.4. Dar de baja unidad motorizada

Cuando una unidad motorizada llega al final de su vida útil, ya sea por razones de desgaste, reparación, destrucción total, entre otros. Se debe marcar dentro del sistema como una unidad de baja, para esto al igual que con el cambio de estatus, el sistema proveerá un control manual con el cual se pondrá a unidad motorizada de baja. Para este cometido debemos localizar el registro de la unidad dentro del listado de unidades motorizadas disponibles, presionar las propiedades del registro y seleccionar Dar de baja, el sistema automáticamente nos mostrara una ventana emergente en donde debemos ingresar el motivo de la baja, finalmente debemos presionar el botón Guardar para que el sistema guarde la información ingresada y asigne el estatus De baja a la unidad motorizada seleccionada.

4.2.3. Catálogo de herramientas y equipo de rescate

El sistema proveerá un catálogo especializado para almacenar la información relacionada con las herramientas que posea la estación de bomberos para las labores de rescate. Este catálogo estará compuesto por los siguientes atributos:

- Tipo de Herramienta
  - Manual
  - Neumática
  - Hidráulica
- Tipo Equipo
  - Autocontenido
  - Casco
  - Casaca contra incendios
  - Pantalón contra incendios
  - Botas contra incendios
  - Guantes
- Propósito
- Estatus
  - Disponible
  - Dañado
  - Prestada
- Origen
  - Compra
  - Donación
  - Préstamo
- Marca
- Modelo

- Número de serie
- Nombre
- Fecha de ingreso
- Fecha de baja
- Razón de baja

4.2.3.1. Agregar herramienta/equipo

Las herramientas y equipos con los que cuenta la estación de bomberos son de vital importancia para llevar a cabo las labores de rescate del día a día, por tal razón debemos mantener el catálogo de estas herramientas de manera efectiva. Para este propósito debemos ingresar las herramientas y equipos, para este propósito debemos presionar el botón Agregar Nuevo, automáticamente el sistema nos mostrara el formulario que debemos llenar con toda la información relacionada a la herramienta o equipo, cabe mencionar que al inicio de este proceso el sistema preguntara si se está ingresando una herramienta o un equipo, esto con el propósito de presentar las propiedades adecuadas en la pantalla de captación de datos. Cuando se termine de ingresar la información asociada a la herramienta o equipo se deberá presionar el botón Guardar para que el sistema grabe la información.

El usuario será redireccionado al listado de herramientas y equipo en el cual podrá ver que el registro recientemente ingresado aparece listado ya dentro de los equipos o herramientas disponibles.

4.2.3.2. Editar herramienta/equipo

Para poder actualizar o editar la información relacionada a una herramienta o equipo se deberán listar los registros disponibles en la pantalla del

catálogo de herramientas y equipo, identificar el registro a editar y presionar sobre las propiedades del registro, debemos entonces seleccionar la opción Editar, automáticamente el sistema mostrará el formulario de registro de herramientas y equipo con la información precargada de la herramienta o equipo. En este punto debemos editar la información que se desea. Al momento de terminar de actualizar la información deseada debemos guardar los cambios para terminar con el proceso, para esto bastará con presionar el botón Guardar para que el sistema proceda a grabar la información registrada.

4.2.3.3. Cambiar estatus herramienta/equipo

Se proveerá de una acción directa desde el catálogo de herramientas y equipo para poder cambiar directamente el estatus asignado. Para este propósito se deberá presionar el botón de Cambiar Estatus que se encontrará en las propiedades de cada registro del listado de herramientas y equipo, se deberá seleccionar el estatus a asignar y presionar el botón Guardar para asignar el nuevo estatus a la herramienta o quipo seleccionado.

4.2.3.4. Dar de baja a herramienta/equipo

Cuando una herramienta o equipo llega al final de su vida útil, ya sea por razones de desgaste, reparación, destrucción total, entre otros. Se debe marcar dentro del sistema como de baja, es decir no disponible. Para este cometido debemos localizar el registro de la herramienta o equipo dentro del listado de disponibles, presionar las propiedades del registro y seleccionar Dar de baja, el sistema automáticamente nos mostrara una ventana emergente en donde debemos ingresar el motivo de la baja, finalmente debemos presionar el botón Guardar para que el sistema guarde la información ingresada y asigne el estatus De baja a la herramienta o equipo seleccionado.

4.2.4. Catálogo de insumos

Los insumos con los que cuentan en la estación de bomberos serán vitales para poder brindar los primeros auxilios a las personas accidentadas o que fueron víctimas de un siniestro. Por insumos podemos mencionar gazas, mariposas, vendas, hilo de sutura, entre otros. Para poder llevar el control de las existencias de cada uno de estos se ha creado el catálogo de insumos.

El catálogo de insumos estará compuesto por los siguientes atributos:

- Nombre
- Tipo
  - Reutilizable
  - Desechable
- Propósito
  - Contener hemorragia
  - Desinfección
  - Limpieza
  - Sutura temporal
  - Desinflamatorio
  - Entre otros...
- Existencia mínima
- Existencia actual

4.2.4.1. Agregar insumo

Para poder agregar un insumo debemos presionar el botón Agregar Nuevo, automáticamente el sistema nos mostrara el formulario que debemos llenar con toda la información relacionada al insumo. Cuando se termine de

ingresar la información asociada al insumo se deberá presionar el botón Guardar para que el sistema grabe la información.

El usuario será redireccionado al listado de insumos, en el cual podrá ver que el registro recientemente ingresado aparece listado ya dentro de los insumos disponibles y su existencia asociada.

4.2.4.2. Editar insumo

Para poder actualizar o editar la información relacionada a un insumo se deberán listar los registros disponibles en la pantalla del catálogo de insumos, identificar el registro a editar y presionar sobre las propiedades del registro, debemos entonces seleccionar la opción Editar, automáticamente el sistema mostrará el formulario de registro del insumo con la información precargada. En este punto debemos editar la información que se desea. Al momento de terminar de actualizar la información requerida debemos guardar los cambios para terminar con el proceso, para esto bastará con presionar el botón Guardar para que el sistema proceda a grabar la información registrada.

4.2.4.3. Actualizar existencia insumo

Cuando un insumo es utilizado o asignado a un botiquín, este presentará un cambio en la existencia del catálogo, por tal razón, se proveerá de una acción directa dentro del catálogo para poder actualizar la existencia asociada. Para este propósito se deberá presionar el botón de Cambiar Existencia que se encontrará en las propiedades de cada registro del listado de insumos, se deberá ingresar la existencia a asignar y presionar el botón Guardar para asignar el nuevo valor de existencia al insumo seleccionado.

4.2.5. Catálogo de servicios/emergencias

La vigésimo novena compañía de Bomberos Voluntarios además de atender las emergencias relacionadas a accidentes y siniestros en la ciudad de Amatitlán, también presta servicios a la comunidad, entre estos podemos mencionar el servicio de traslado de enfermos del Hospital Nacional de Amatitlán hacia otro centro asistencial, el llenado de depósitos de agua utilizando la motobomba de la estación, entre otros. Para poder llevar el control de estos servicios o emergencias vamos a utilizar el catálogo de servicios y emergencias, en el cual vamos a registrar todos los tipos de emergencias y servicios que se presentan en el día a día de la compañía.

Este catálogo estará compuesto por los siguientes atributos:

- Nombre
- Categoría
    - Servicio
        - Traslado de paciente
        - Llenado de cisterna
        - Guardia de rescate en eventos
        - Entre otros
    - Emergencia
        - Incendio
        - Accidente de transito
        - Enfermedad común
        - Herido de bala
        - Herido por elementos punzo cortantes
        - Intoxicación/envenenamiento
        - Maternidad

- Choque eléctrico
- Quemaduras
- Accidente colectivo
- Amputación
- Entre otros

- Unidad motorizada (referencia a un tipo de unidad motorizada que es requerido para brindar el servicio o atender la emergencia)

4.2.5.1. Ingresar servicio/emergencia

Para agregar una emergencia o servicio al catálogo se debe presionar el botón Agregar Nuevo, automáticamente el sistema nos mostrara el formulario que debemos llenar con toda la información relacionada al servicio o emergencia. Cuando se termine de ingresar la información asociada al servicio o emergencia se deberá presionar el botón Guardar para que el sistema grabe la información.

El usuario será redireccionado al listado de servicios/emergencias, en el cual podrá ver que el registro recientemente ingresado aparece listado ya dentro de los servicios/emergencias disponibles con su información más relevante asociada.

4.2.5.2. Editar servicio/emergencia

Para poder actualizar o editar la información relacionada a un servicio o emergencia se deberán listar los registros disponibles en la pantalla del catálogo de servicios/emergencias, debemos identificar el registro a editar y presionar sobre las propiedades del registro, debemos entonces seleccionar la opción Editar, automáticamente el sistema mostrará el formulario de registro del servicio

o emergencia con la información precargada. En este punto debemos editar la información que debamos actualizar. Al momento de terminar de actualizar la información requerida, debemos guardar los cambios para terminar con el proceso, para esto bastará con presionar el botón Guardar para que el sistema proceda a grabar la información recién editada.

4.2.5.3. Eliminar servicio/emergencia

En el caso de que se desee eliminar un tipo de emergencia o servicio del catálogo, esto se podrá realizar utilizando las opciones de las propiedades asociadas al registro. Para poder llevar acabo esta acción, debemos listar los servicios/emergencias disponibles, ubicamos el registro que deseamos eliminar, presionamos sobre las propiedades del registro y seleccionamos Eliminar. Automáticamente el sistema nos presentara una alerta indicando que el registro se eliminara, debemos confirmar la acción presionando el botón Si. El registro se eliminará del catálogo, lo cual vamos a poder evidenciar listando nuevamente los servicios/emergencias disponibles, el registro eliminado ya no estará más disponible dentro del catálogo.

4.3. Menú operaciones

En este menú vamos a tener las pantallas operativas del sistema, estas pantallas estarán alimentadas por los catálogos que anteriormente definidos. El propósito de estas pantallas es realizar los procesos de registro de llamadas, las cuales en su mayoría serán para el reporte de emergencias, además de registrar también las llamadas de solicitud de servicios varios.

Dentro de las operaciones también se podrán generar los reportes asociados a las llamadas recibidas y atendidas, esto con el objetivo de tener estadísticas que ayuden a la compañía de Bomberos Voluntarios.

4.3.1. Registro de llamadas de emergencia

Las llamadas que ingresen a la cabina de la estación de bomberos deberán ser registradas con el procedimiento descrito a continuación.

4.3.1.1. Ingresar llamada de emergencia

El telefonista de turno atenderá una llamada entrante a la compañía de Bomberos Voluntarios. Para esto el telefonista se deberá dirigir al menú de operaciones y seleccionar Ingreso de Llamadas de Emergencia. En esta pantalla deberá ingresar el tipo de emergencia reportado, la dirección de donde se requiere la atención, un punto de referencia adicional a la dirección, el nombre de la persona que reporta la emergencia, entre otros.

La llamada será ingresada al sistema para posteriormente proceder con la asignación de bomberos y la unidad motorizada que atenderá la emergencia, este proceso se detallará a continuación.

4.3.1.2. Editar llamada de emergencia

Si el telefonista de turno requiere editar o actualizar algún dato de la llamada de emergencia ingresada anteriormente, deberá dirigirse a la pantalla del listado de llamadas de emergencia, ubicar el registro y seleccionar las propiedades de este, seleccionar Editar llamada, automáticamente se desplegará el formulario con la información precargada de la emergencia. Seleccionar los

campos que es necesario editar o actualizar, seguidamente se deberá presionar el botón Guardar para que el sistema grabe la información.

4.3.1.3. Búsqueda de llamadas de emergencia

Un proceso vital para la compañía de bomberos es la ubicación de manera eficiente de un registro de llamada de emergencia, para este cometido se dotará al sistema con filtros de búsqueda especializados, los cuales ayudaran al usuario a encontrar de manera rápida un registro especifico.

Los filtros disponibles serán los siguientes:

- Fecha de la llamada
- Dirección de atención de emergencia (filtro con condición contiene para incluir todas las referencias de una dirección o punto de referencia)
- Unidad motorizada asignada
- Bomberos asignados
- Persona que reporta
- Número de teléfono de donde se reportó la emergencia

4.3.2. Asignación de unidades motorizadas

Para atender una llamada de emergencia es necesario el registro de la llamada como vimos en la sección anterior, este será el primer paso, el siguiente paso es asignar la unidad motorizada con la cual se atenderá la emergencia reportada, dependiendo del tipo de emergencia así se designará la unidad motorizada idónea para proceder con la labor de rescate.

4.3.2.1. Asignar unidad motorizada

Para este cometido se deberá buscar la emergencia o servicio dentro del listado de emergencias y servicios desplegado en la pantalla operativa, la última emergencia o servicio recibido será la que se muestra al inicio de esta lista, las demás emergencias se listaran en forma descendente conforme la hora y fecha de recepción. Se deberá presionar sobre las propiedades de la llamada y seleccionar Asignar Unidad, al momento de realizar esta acción se desplegará un listado de las unidades disponibles en la compañía para poder asignar, estas se podrán filtrar por el tipo de emergencias si el operador de cabina necesita asignar una unidad específica, por ejemplo, una motobomba para una emergencia de tipo incendio.

Cabe resaltar que para una emergencia se podrán asignar más de una unidad motorizada cuando la emergencia así lo requiera.

4.3.2.2. Asignar bomberos

Una emergencia o servicio debe tener asignados también los bomberos que serán designados para atender la emergencia, para realizar esta acción se deberá ubicar sobre la llamada de emergencia en el listado de la pantalla operativa, seleccionar las propiedades del registro y presionar sobre Asignar bombero, automáticamente el sistema presentará un listado elegible de bomberos disponibles en la estación. Como en el caso de las unidades motorizadas, a una emergencia se le podrán asignar más de un bombero cuando la emergencia así lo requiera.

4.3.2.3. Editar asignación

En la operación de las llamadas de emergencia o servicios, en algunas ocasiones se deberá actualizar las asignaciones tanto de unidades motorizadas como de bomberos, estas actualizaciones se pueden presentar por temas de errores humanos o de cambios de última hora. Para llevar a cabo esta acción se deberá dirigir a la página operativa del control de llamadas, ubicar la emergencia a editar, seleccionar sus propiedades y presionar sobre Editar Asignaciones, automáticamente el sistema mostrara las asignaciones de bomberos y unidades motorizadas, se deberán realizar los cambios que se necesiten (eliminar o agregar nuevos registros de bomberos y unidades motorizadas), por último se debe presionar el botón Guardar para que el sistema grabe la información ingresada.

4.3.3. Menú reportes

Dentro de las operaciones de la compañía, es de vital importancia la información histórica que se va generando día con día en la estación de bomberos, por esta razón es importante el poder generar resúmenes que nos den estadísticas de lo que se ha operado en cuento a emergencias y servicios se refiere. Estos resúmenes serán los reportes operativos que el sistema será capaz de generar en el momento que los usuarios los soliciten.

4.3.3.1. Reporte de emergencias

Este reporte tiene como objetivo el listar de manera clara y ordenada las emergencias atendidas durante un periodo requerido. El reporte se podrá generar utilizando los siguientes criterios:

- Fecha de inicio
- Fecha final
- Tipo de emergencia
- Dirección (criterio a ingresar podría ser el nombre de una colonia, barrio, cantón, entre otros)
- Entre otros

Mientras más criterios de filtrado sean seleccionados, la información en el reporte será menos general y más personalizada. Para generar el reporte el sistema solicitara ingresar al menos las fechas de inicio y final para extraer la información de la base de datos, este filtro de fechas será obligatorio, esto para poder prevenir que la extracción masiva de datos pueda afectar el rendimiento de la base de datos y de la aplicación web.

Una vez generado el reporte, se podrá visualizar en primera instancia en pantalla, pero cuando se necesite presentar o enviar el informe por correo electrónico o físicamente, este se podrá exportar a un documento en formato PDF, este formato es ligero para poder enviarlo mediante correo electrónico o poder imprimirlo para poderlo presentar físicamente.

4.3.3.2. Reporte de servicios

Los servicios varios prestados por la compañía de bomberos como apoyo a la comunidad también se deben de poder visualizar de manera resumida, ya que es de vital importancia para la junta de oficiales el saber en dónde está la mayor demanda de servicios de la población, así como poder administrar de una manera óptima los recursos necesarios para poder brindar a la población un servicio eficiente.

Este resumen lo vamos a poder generar mediante un reporte, similar al de emergencias, este reporte también podrá ser filtrado por varios criterios, los cuales se listan a continuación:

- Fecha de inicio
- Fecha final
- Tipo de servicio
- Dirección (criterio a ingresar podría ser el nombre de una colonia, barrio, cantón, entre otros)
- Entre otros

Para generar el reporte el sistema solicitara ingresar al menos las fechas de inicio y final para extraer la información de la base de datos, este filtro de fechas será obligatorio, esto para poder prevenir que la extracción masiva de datos pueda afectar el rendimiento de la base de datos y de la aplicación web.

El reporte se podrá visualizar en primera instancia en pantalla, pero cuando se necesite presentar o enviar el informe por correo electrónico o físicamente, este se podrá exportar a un documento en formato PDF.

5. *SOFTWARE* DESARROLLADO

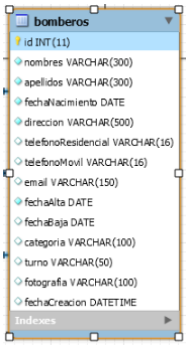
Durante la etapa de desarrollo de *software*, se tomaron en cuenta todos los detalles de funcionalidad en cuanto a la operatividad de la compañía de bomberos se refiere. Vamos pues a describir cada uno de los componentes desarrollados durante esta fase.

5.1. Diseño de base de datos

Para poder cumplir con el propósito de la aplicación definimos el siguiente modelo de datos, el cual consta de 20 tablas o entidades dentro del esquema:

- Bomberos: tabla en donde se almacenarán los datos concernientes a un bombero dentro de la estación, algunos de los datos relevantes son el nombre, apellidos, fecha de nacimiento, dirección, teléfono móvil, fecha de alta, entre otros.

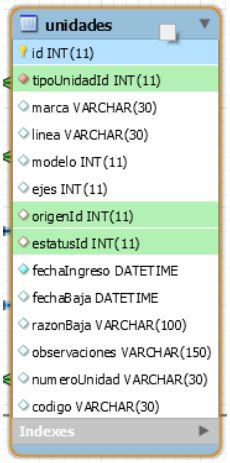
Figura 12. **Diseño de tabla bomberos**



Fuente: elaboración propia, realizado con MySql Workbench.

- Unidades: tabla en la cual se almacenan las unidades de rescate que pertenecen a la estación de bomberos, se registrará la marca de la unidad, el modelo, la línea, cantidad de ejes, su origen, estatus, fecha de ingreso, entre otros.

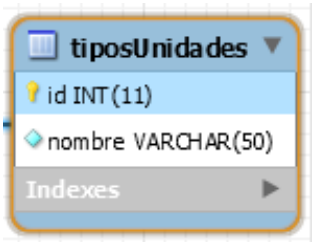
Figura 13. **Diseño de tabla unidades**



Fuente: elaboración propia, realizado con MySql Workbench.

- Tipos unidades: tabla que almacena los diferentes tipos de unidades que pueden ser parte de la compañía de bomberos, por ejemplo, panel, *pick up*, camión, cabezal, entre otros. Las llaves primarias de esta tabla serán llaves foráneas en la tabla unidades para designar un tipo de unidad por cada unidad.

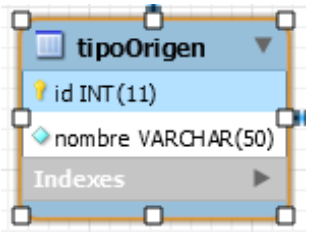
Figura 14. **Diseño de tabla tipos unidades**



Fuente: elaboración propia, realizado con MySql Workbench.

- Tipo origen: tabla donde se almacenan los valores asociados a los tipos de origen de una unidad o insumo, los tipos de origen pueden ser donaciones, compras directas, préstamo, entre otros.

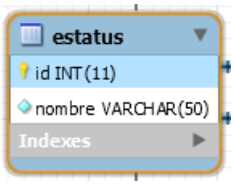
Figura 15. **Diseño de tabla tipo origen**



Fuente: elaboración propia, realizado con MySql Workbench.

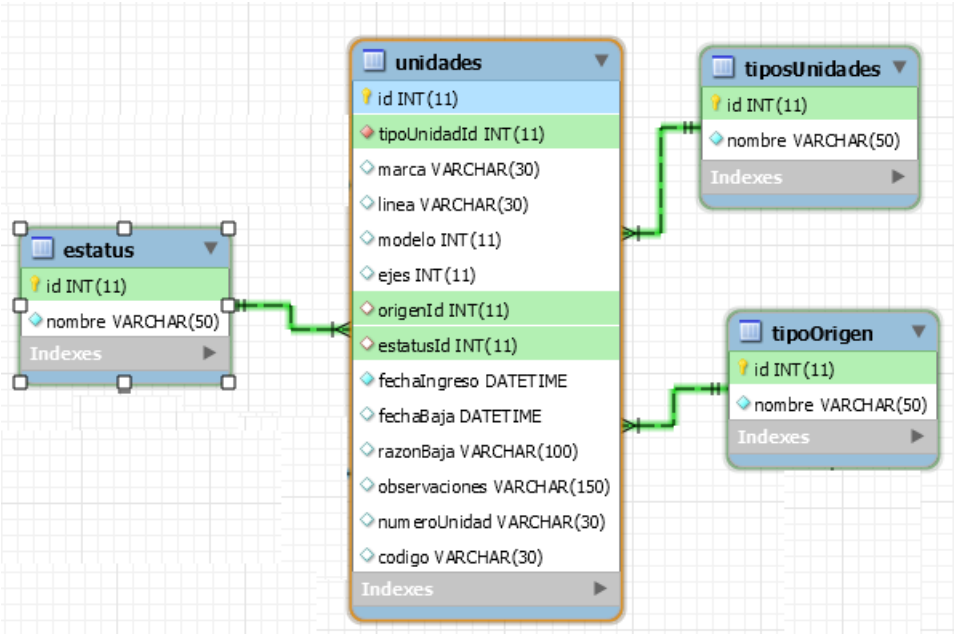
- Estatus: tabla donde se almacenan los diferentes estatus que podrán tener las unidades, insumos o herramientas dentro de la compañía de bomberos.

Figura 16. **Diseño de tabla estatus**



Fuente: elaboración propia, realizado con MySql Workbench.

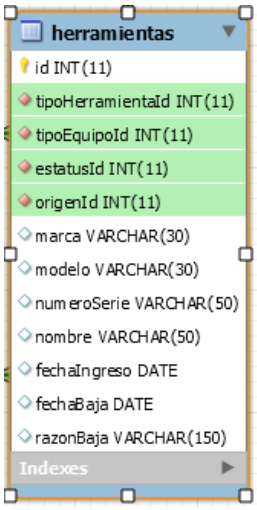
Figura 17. **Entidad relación unidades de rescate**



Fuente: elaboración propia, realizado con MySql Workbench.

- Herramientas: tabla en donde se almacenarán las herramientas que se quieran registrar en la estación de bomberos para las labores de rescate, estas contarán con una referencia hacia estatus, tipo de herramienta, tipo equipo, origen, entre otros.

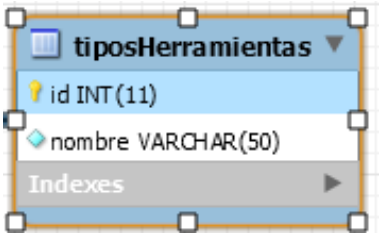
Figura 18. **Diseño de tabla herramientas**



Fuente: elaboración propia, realizado con MySql Workbench.

- Tipos herramientas: tabla en donde se almacenarán los diferentes tipos de herramientas, pueden ser neumáticas, manuales, hidráulicas, entre otros.

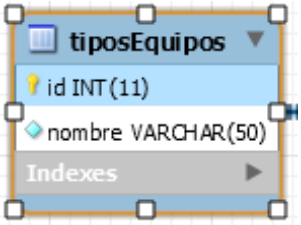
Figura 19. **Diseño de tabla tipos herramientas**



Fuente: elaboración propia, realizado con MySql Workbench.

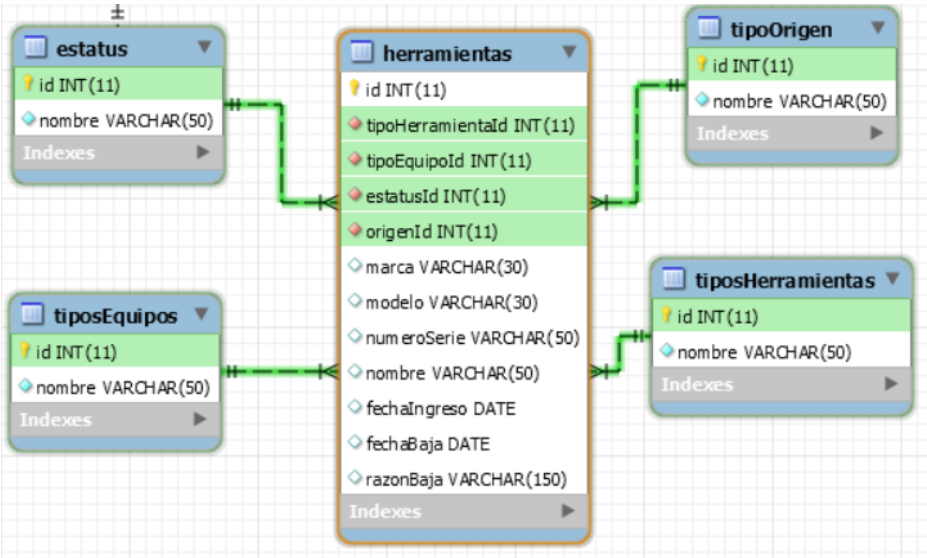
- Tipos equipo: tabla en donde se guardarán los diferentes tipos de equipo a la que puede pertenecer una herramienta, pudiendo ser casco, autocontenido, casaca contra incendios, entre otros.

Figura 20. **Diseño de tabla tipos equipo**



Fuente: elaboración propia, realizado con MySql Workbench.

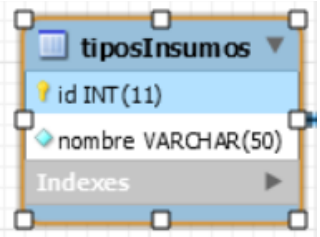
Figura 21. **Entidad relación herramientas**



Fuente: elaboración propia, realizado con MySql Workbench.

- Tipos insumos: tabla que almacenara los diferentes tipos de insumos que son parte de la operación diaria de la estación de bomberos, estos pueden ser reutilizables, desechables, entre otros.

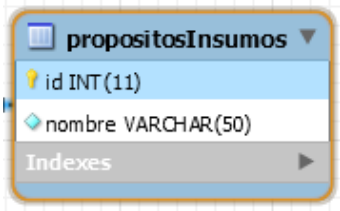
Figura 22. **Diseño de tabla tipos insumos**



Fuente: elaboración propia, realizado con MySql Workbench.

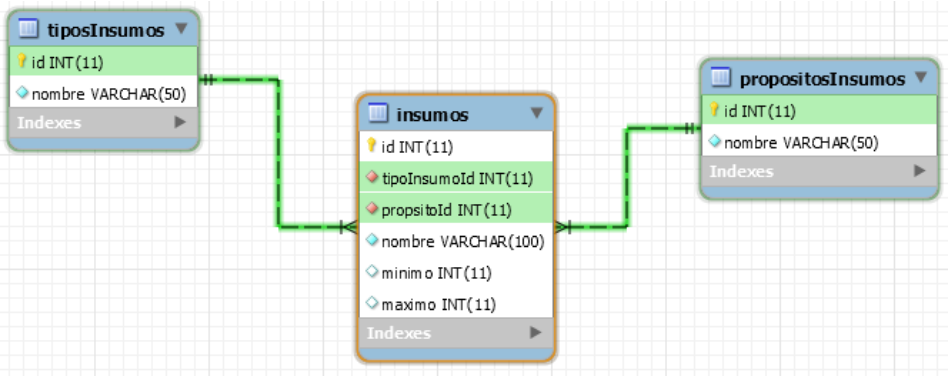
- Propósitos insumos: tabla para almacenar los diferentes propósitos para los que un insumo puede servir dentro de la operación de la estación de bomberos, algunos de estos serán contener hemorragias, inmovilizar miembros, desinfección, entre otros.

Figura 23. **Diseño de tabla propósitos insumos**



Fuente: elaboración propia, realizado con MySql Workbench.

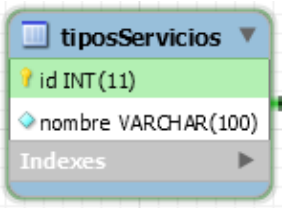
Figura 24. **Entidad relación insumos**



Fuente: elaboración propia, realizado con MySql Workbench.

- Tipos servicios: tabla en donde se guardarán los tipos de servicios que presta la estación de bomberos, los tipos servicios comunes pueden ser llenado de cisternas particulares, traslado de pacientes, bomberos de turno para cubrir eventos, grúa, entre otros.

Figura 25. **Diseño de tabla tipos de servicios**

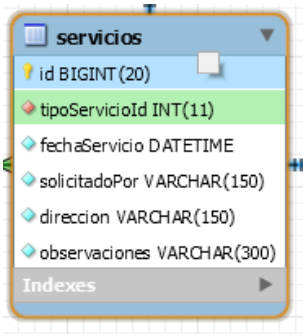


Fuente: elaboración propia, realizado con MySql Workbench.

- Servicios: tabla en donde se almacenarán todos los registros concernientes a los servicios que se realicen en la estación de bomberos para la población en general, en esta tabla se podrá registrar la fecha de

cuando se realizó el servicio, el nombre del servicio, la persona que lo solicitó, la dirección donde se realizó, entre otros.

Figura 26. **Diseño de tabla de servicios**



Fuente: elaboración propia, realizado con MySql Workbench.

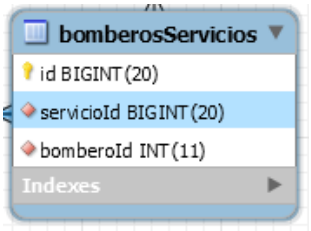
- Unidades servicios: tabla utilizada como detalle para los servicios prestados, en esta tabla se almacenarán las unidades de rescate utilizadas en un servicio determinado, un servicio puede estar relacionado a varias unidades de rescate.

Figura 27. **Diseño de tabla de unidades servicios**

Fuente: elaboración propia, realizado con MySql Workbench.

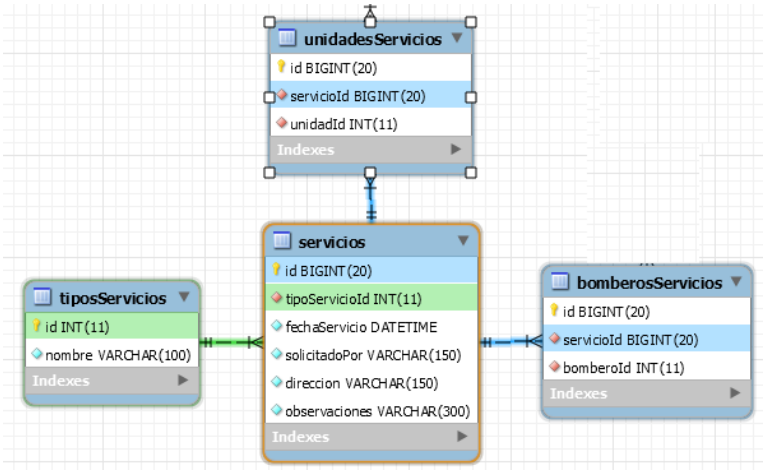
- Bomberos servicios: tabla que servirá como detalle para servicios, en esta tabla se almacenaran los bomberos que atendieron un servicio en específico, ya que un servicio podrá involucrar varios bomberos para llevarse a cabo.

Figura 28. **Diseño de tabla de unidades servicios**



Fuente: elaboración propia, realizado con MySql Workbench.

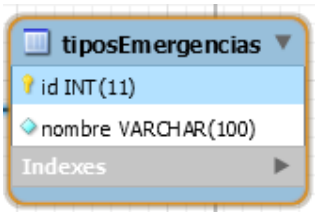
Figura 29. **Entidad relación servicios**



Fuente: elaboración propia, realizado con MySql Workbench.

- Tipos emergencias: tabla en donde se almacenarán los diferentes tipos de emergencias que son atendidas a diario por la compañía de bomberos. Podemos mencionar enfermedad común, herido de bala, herido con arma punzo cortante, choque eléctrico, covid, entre otros.

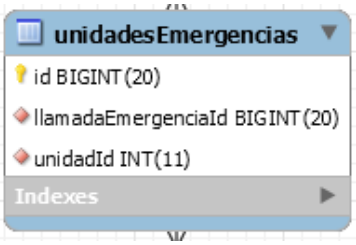
Figura 30. **Diseño de tabla de tipos emergencias**



Fuente: elaboración propia, realizado con MySql Workbench.

- Unidades emergencias: tabla utilizada como detalle para las emergencias atendidas, en esta tabla se almacenarán las unidades de rescate utilizadas en una emergencia determinada, una emergencia puede estar relacionada a varias unidades de rescate.

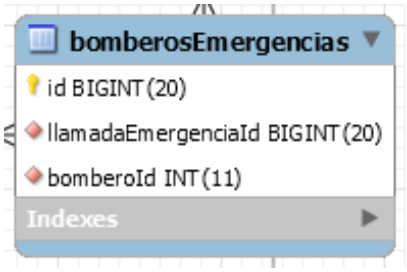
Figura 31. **Diseño de tabla de unidades emergencias**



Fuente: elaboración propia, realizado con MySql Workbench.

- Bomberos emergencias: tabla que servirá como detalle para emergencias, en esta tabla se almacenaran los bomberos que atendieron una emergencia en específico, ya que una emergencia podrá ser atendida por varios bomberos.

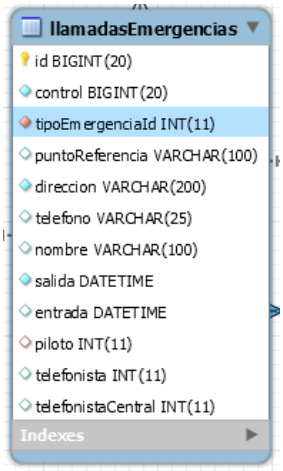
Figura 32. **Diseño de tabla de bomberos emergencias**



Fuente: elaboración propia, realizado con MySql Workbench.

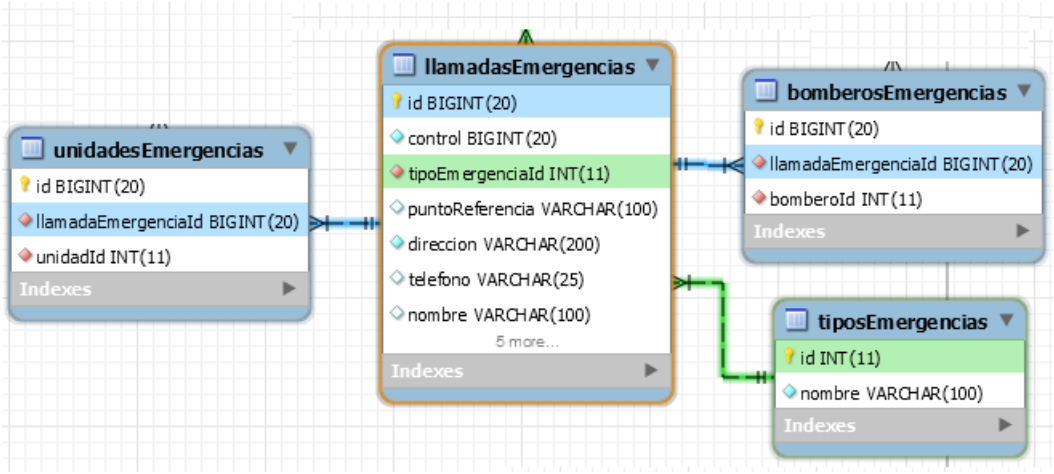
- Llamadas emergencias: tabla en donde se almacenarán los registros concernientes a las llamadas de emergencia que entran a la estación de bomberos, estas emergencias constaran de un tipo de emergencia, dirección de donde se suscita, persona que la reporta, entre otros.

Figura 33. **Diseño de tabla de llamadas emergencias**



Fuente: elaboración propia, realizado con MySql Workbench.

Figura 34. **Entidad relación emergencias**



Fuente: elaboración propia, realizado con MySql Workbench.

5.2. Diseño del API

Como se estableció en capítulos anteriores, el API que se desarrolló es del tipo RESTful, se desarrolló en el lenguaje PHP con un *framework* especializado para el mapeo de las entidades relacionales desde la base de datos. A continuación, vamos a detallar cada uno de los componentes de *software* que conforman esta API.

5.2.1. Lumen *framework*

El *framework* LUMEN es una derivación del ya conocido LARAVEL, al referirnos como una derivación queremos decir que es una implementación de este marco de trabajo, pero con la singularidad que es más liviano, está diseñado para aplicaciones que necesitan un performance rápido y ágil, lo cual es primordial para una estación de bomberos que cubre emergencias 24 horas 7 días a la semana.

LUMEN provee una plataforma ágil de desarrollo de RESTful APIs orientadas a microservicios, es decir, orientado a la modularidad, segmentos de código enteros que se dedican a una sola tarea.

5.2.2. Eloquent *framework*

LUMEN utiliza un *software* ORM para poder trabajar con un modelo de datos contenido dentro una base de datos, también provee la capacidad de crear este modelo en caso no exista y nuestras necesidades requieran un alto acoplamiento con la herramienta, para este caso, se utilizó Eloquent con un modelo de datos ya establecido.

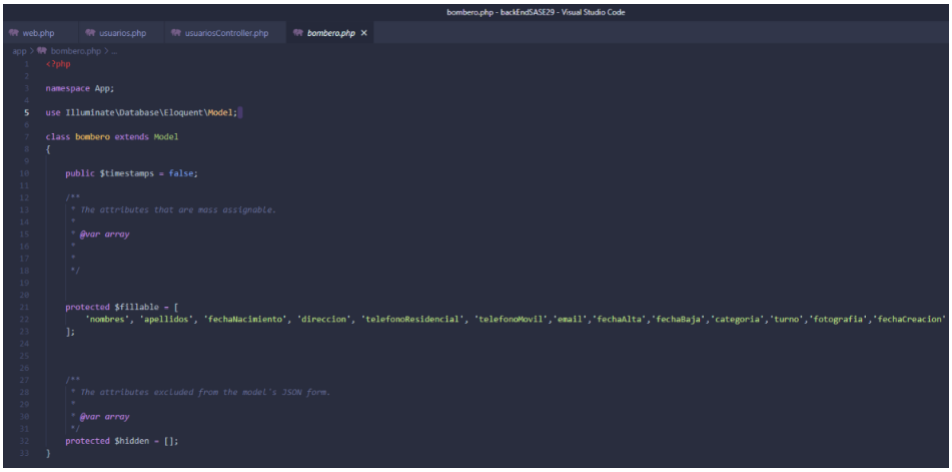
ORM (*Object Relational Mapping*) es un *framework* que se encarga de realizar el mapeo de cada entidad o tabla desde una base datos, hacia una clase dentro de nuestro código fuente, cada una de estas clases corresponderá a una tabla dentro del modelo de datos. Esta será la función de Eloquent, mapear cada una de las tablas del modelo de datos hacia nuestra API como una clase instanciable, en la cual vamos a definir sus atributos y relaciones con otras tablas dentro del modelo.

A continuación, se mostrará como se definieron los componentes del API.

5.2.3. Modelos

Se definieron modelos de datos como clases dentro del API para poder mapear por medio de Eloquent cada una de las entidades o tablas de la base de datos, en la figura a continuación se muestra un modelo bombero.php, el cual contiene la definición de la tabla bomberos.

Figura 35. **Modelo bombero.php**



Fuente: elaboración propia, realizado con Visual studio Code.

En la figura anterior podemos evidencias que el modelo contiene específicamente los campos pertenecientes a la entidad bomberos en la base de datos, cabe mencionar que no se define el campo id ya que este valor es una secuencia automática dentro de la base de datos, por ende, no se debe declarar dentro del arreglo *fillable* ya que como su nombre lo indica, únicamente debemos especificar los campos que pueden ser llenados por el API.

En el caso se quieran ocultar algunos campos con información sensible al usuario, podemos hacer uso del arreglo *hidden* en el cual podemos definir que campos será ocultos a la vista del usuario.

En el API desarrollada se definieron 20 modelos, en donde cada uno corresponde a una entidad o tabla de la base de datos definida con anterioridad. En la tabla siguiente se listan cada uno de estos modelos:

Tabla III. **Modelos desarrollados por entidad**

Modelo	Entidad
bombero.php	bomberos
bombero.php	bomberosEmergencias
bomberosServicios.php	bomberosSercvicios
estatus.php	estatus
herramientas.php	herramientas
insumos.php	insumos
llamadasEmergencias.php	llamadasEmergencias
propositosInsumos.php	propositosInsumos
servicios.php	servicios
tipoOrigen.php	tipoOrigen
tiposEmergencias.php	tiposEmergencias
tiposEquipos.php	tiposEquipo
tiposHerramientas.php	tiposHerramientas
tiposInsumos.php	tiposInsumos
tiposServicios.php	tiposSevicios
tiposUnidades.php	tiposUnidades
unidades.php	unidades
unidadesEmergencias.php	unidadesEmergencias
unidadesServicios.php	unidadesServicios
usuarios.php	usuarios

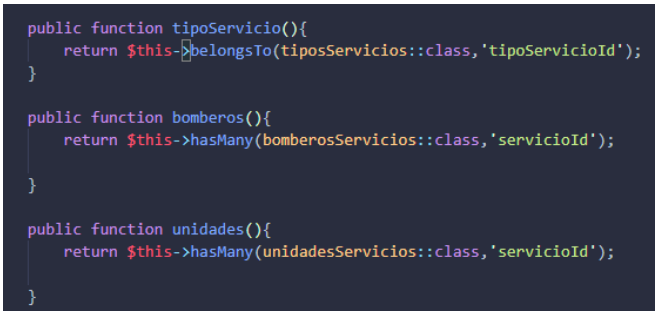
Fuente: SASE29 (2021). *Modelos y entidades asociadas.* Consultado el 10 de octubre de 2021. Recuperado de código fuente propio de sistema SASE29.

5.2.4. Relaciones entre modelos

Cada uno de los modelos creados debe obedecer en su definición fielmente al modelo de datos que se está mapeando desde la base de datos en MySql, en la base de datos hemos definido diferentes tipos de relaciones, relaciones uno a uno, uno a muchos, entre otras. Por tal razón debemos indicar estas relaciones dentro de nuestro modelo en el API.

En la siguiente figura podemos evidenciar la definición de estas relaciones, para este cometido vamos a utilizar como ejemplo el modelo de servicios:

Figura 36. **Relaciones del modelo servicios**



```
public function tipoServicio(){
    return $this->belongsTo(tiposServicios::class,'tipoServicioId');
}

public function bomberos(){
    return $this->hasMany(bomberosServicios::class,'servicioId');

}

public function unidades(){
    return $this->hasMany(unidadesServicios::class,'servicioId');

}
```

Fuente: elaboración propia, realizado con Visual Studio Code.

En la figura anterior podemos ver que las relaciones dentro de los diferentes modelos se definen como funciones públicas que retornan la instancia del modelo que corresponde según el tipo de relación:

- Relación uno a uno: esta relación se definirá con la instrucción belongsTo la cual indica que la entidad servicios (actual) le pertenece un registro del modelo tiposServicios.php, ya que un servicio tiene un tipo de servicio asociado, únicamente uno, no puede tener varios.

- Relación uno a muchos: este tipo de relación la vamos a definir con la instrucción hasMany la cual indicara al *software* de Eloquent que la entidad servicios (actual) contiene un conjunto de modelos relacionados del modelo bomberosServicios.php, ya que en este modelo como explicamos en el capítulo anterior, se encarga de guardar a los bomberos que atendieron un determinado servicio, dado que varios bomberos pueden

atender un servicio, la relación uno a muchos significa en este caso que se tendrá un conjunto indeterminado de bomberos asociados.

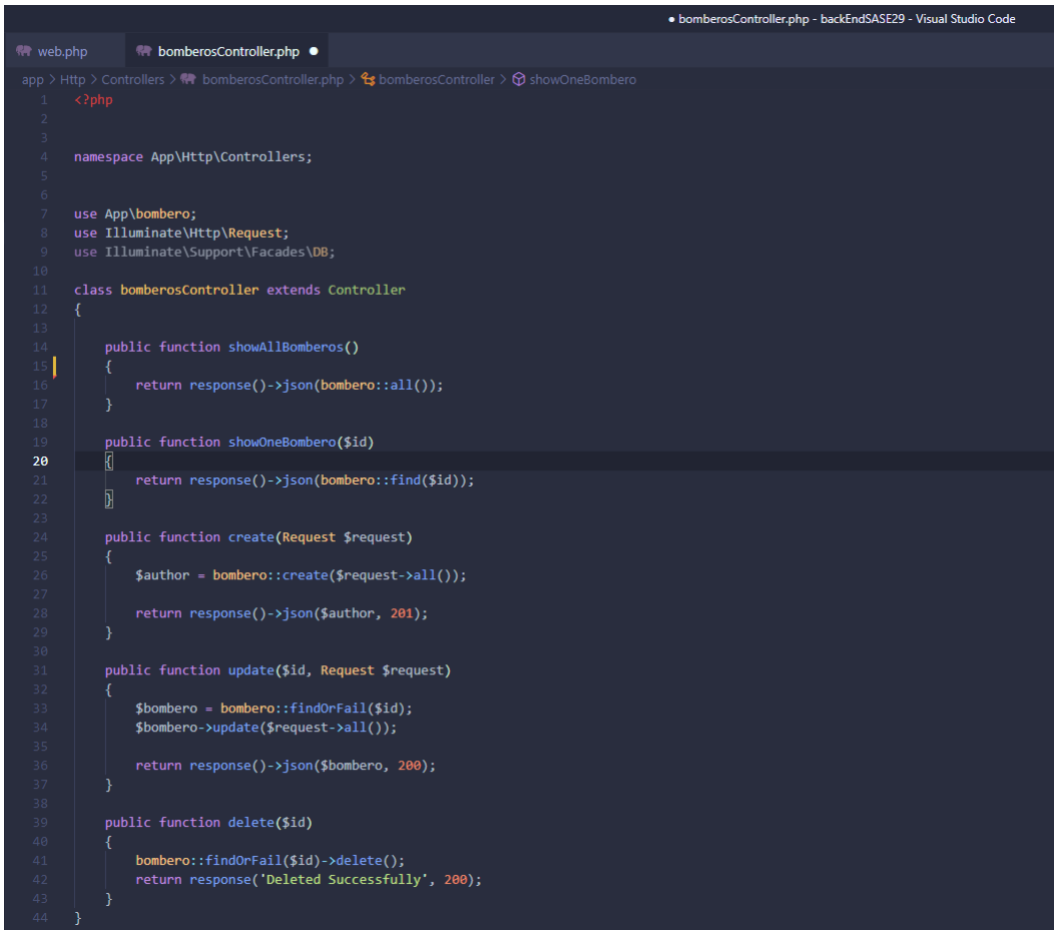
5.2.5. Controladores

Los modelos que definimos en la sección anterior necesitaran un controlador que se encargue de utilizar los modelos definidos cuando la API sea invocada por parte de peticiones de usuarios de la aplicación web. Estos controladores se definen por cada modelo, es decir, no hay modelo que no tenga un controlador asociado.

Dentro de estos controladores vamos a definir los métodos que se ejecutaran según sea el tipo de petición HTTP, recordemos que en el capítulo 2 hablamos acerca de estas peticiones *get, post, put*, entre otros.

En la siguiente figura vamos a poder evidenciar el contenido de estos controladores, para este cometido utilizaremos el controlador de bomberos como ejemplo:

Figura 37. **Controlador bomberos Controller.php**



Fuente: elaboración propia, realizado con Visual Studio Code.

Como podemos observar en la figura anterior, el controlador es de tipo HTTP, es decir va a responder a ante una petición de este tipo. Cada controlador tiene varios métodos, los cuales se van a ejecutar, como ya hemos dicho, según la petición realizada por el usuario:

- Método *showall*<modelo>: este método se encargará de mostrar todos los bomberos que se encuentran en la tabla bomberos dentro de la base de datos, este método se ejecutara con una petición HTTP *get*.

- Método *showone*<modelo>: este método se encarga de mostrar un bombero, para lo cual en la petición se debe incluir el identificador único del bombero dentro de la base de datos (*id*). Se ejecutará con una petición HTTP *get/id*.

- Método *create*: este método será el encargado de agregar un nuevo registro a la tabla bomberos dentro de la base de datos, en otras palabras, se encargará de guardar un nuevo bombero, se ejecutará con una petición HTTP *post*.

- Método *update*: este método se encargará de realizar una actualización sobre una entidad bombero, es decir, vamos a poder cambiar o actualizar el valor de una propiedad de este modelo. Se ejecutará con una petición HTTP *put/id*, ya que debemos indicar el identificador único del modelo para que pueda ser actualizado.

- Método *delete*: se encargará de eliminar un registro de bombero dentro del sistema y la base de datos, debemos indicar el identificador único del registro, se ejecutará con una petición HTTP *delete/id*.

En la siguiente tabla vamos a listar los controladores que se desarrollaron para poder responder a las peticiones de clientes desde la aplicación web:

Tabla IV. **Controladores por modelo asociado**

Controlador	Modelo Asociado
bomberoController.php	bombero.php
bomberosEmergenciasController.php	bomberosEmergencias.php
bomberosServiciosController.php	bomberosServicios.php
estatusController.php	estatus.php
herramientasController.php	herramientas.php
insumosController.php	insumos.php
llamadasEmergenciasController.php	llamadasEmergencias.php
propositosInsumosController.php	propositosInsumos.php
serviciosController.php	servicios.php
tipoOrigenController.php	tipoOrigen.php
tiposEmergenciasController.php	tiposEmergencias.php
tiposEquiposController.php	tiposEquipos.php
tiposHerramientasController.php	tiposHerramientas.php
tiposInsumosController.php	tiposInsumos.php
tiposServiciosController.php	tiposServicios.php
tiposUnidadesController.php	tiposUnidades.php
unidadesController.php	unidades.php
unidadesEmergenciasController.php	unidadesEmergencias.php
unidadesServiciosController.php	unidadesServicios.php
usuariosController.php	usuarios.php

Fuente: SASE29 (2021). *Controladores y modelos asociados.* Consultado el 10 de octubre de 2021. Recuperado de código fuente propio de sistema SASE29.

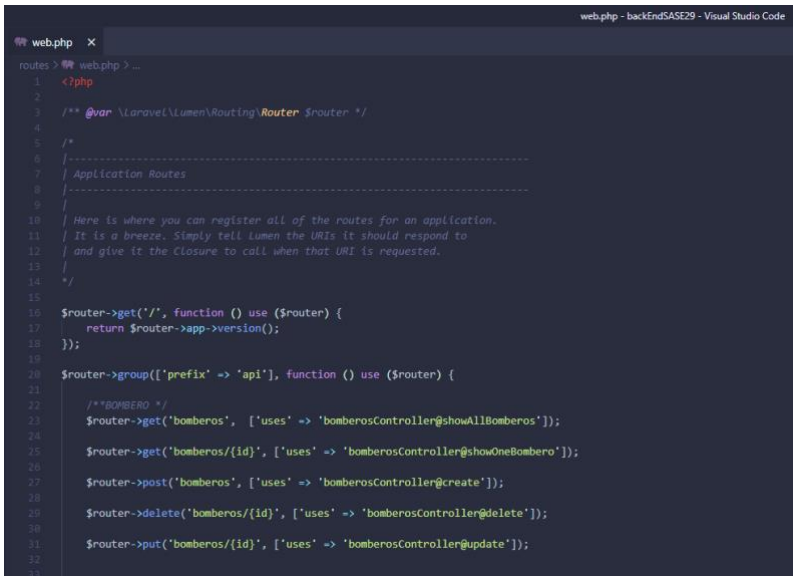
5.2.6. Rutas

El API entrara en acción cuando sea consumida desde un cliente web, en este caso nuestro cliente es la aplicación web, para este cometido, la aplicación utilizara diferentes tipos de peticiones HTTP, como hemos hablado con anterioridad, estas peticiones son básicamente *get, post, put, delete*, entre otras. Por esta razón debemos definir dentro de nuestra API, las llamadas rutas las

cuales son por decirlo de alguna manera, los conductos o caminos por los cuales puede ser invocada o consumida nuestra API.

Las rutas obedecerán al tipo de petición HTTP según sea el caso, vamos a evidenciar esto en la siguiente figura, en la cual veremos cómo se definen las rutas para poder invocar al controlador de bomberos. Dicho esto, debemos aclarar que cada ruta definida dentro del API debe corresponder a un controlador, ya que al momento de que nuestra API sea consumida, esta debe saber a qué controlador llamar según sea la ruta, además de indicar el tipo de petición HTTP para saber que método ejecutar dentro del controlador.

Figura 38. **Rutas de controlador bomberos**



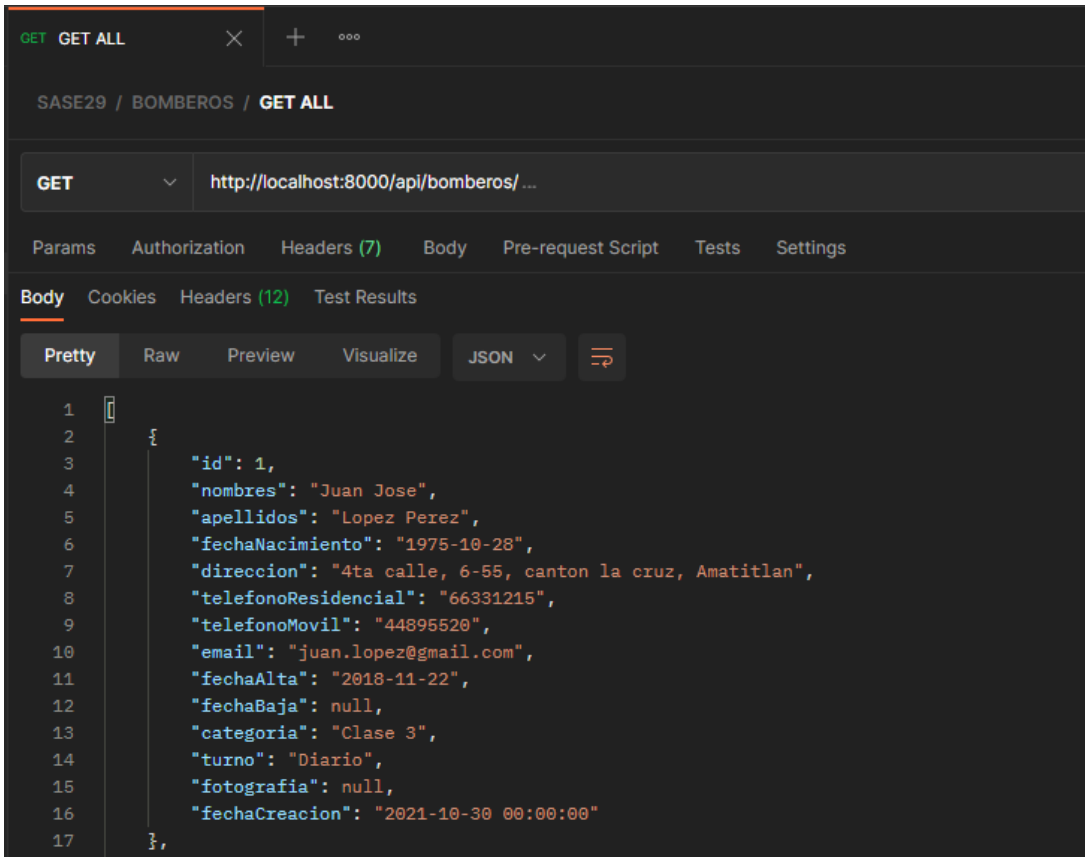
Fuente: elaboración propia, realizado con Visual Studio Code.

En la figura anterior vemos como cada ruta del controlador de bomberos está definida para cada tipo de petición HTTP. Por ejemplo, la ruta de petición *put* invocará al controlador de bomberos indicándole que debe ejecutar el método

update, para el caso de la petición *get*, se indica que debe invocar al controlador de bomberos y este a su vez debe ejecutar el método *showAllBomberos*.

En la siguiente figura se muestra una petición *get* realizada desde un cliente, veremos cómo se invoca la ruta y el resultado de la petición:

Figura 39. **Petición *get* a la ruta de bomberos**



Fuente: elaboración propia, realizado con Postman.

Como podemos evidenciar en la figura anterior, vemos que el API está escuchando las peticiones en el puerto 8000, además vemos la ruta definida para el controlador de bomberos y el tipo de petición se configuro para que fuera HTTP

get. El API al momento de detectar la ruta y el tipo de petición, ejecuto el controlador de bomberos y este a su vez por el tipo de petición invoco el método que devuelve un código de estatus 200(ok) y el listado de bomberos en el sistema.

En la siguiente tabla vamos a listar todas las rutas definidas dentro de la API, las peticiones HTTP que aceptan y el controlador asociado a cada ruta definida:

Tabla V. **Rutas por modelo y controlador asociado**

Ruta	Peticiones	Controlador
http://localhost:8000/api/bomberos/	*get, post, put, delete*	bomberoController.php
http://localhost:8000/api/tiposUnidades/	*get, post, put, delete*	tiposUnidadesController.php
http://localhost:8000/api/tipoOrigen	*get, post, put, delete*	tipoOrigenController.php
http://localhost:8000/api/estatus	*get, post, put, delete*	estatusController.php
http://localhost:8000/api/unidades/	*get, post, put, delete*	unidadesController.php
http://localhost:8000/api/tiposHerramientas/	*get, post, put, delete*	tiposHerramientasController.php
http://localhost:8000/api/tiposEquipos/	*get, post, put, delete*	tiposEquiposController.php
http://localhost:8000/api/herramientas/	*get, post, put, delete*	herramientasController.php
http://localhost:8000/api/tiposInsumos/	*get, post, put, delete*	tiposInsumosController.php
http://localhost:8000/api/propositosInsumos/	*get, post, put, delete*	propositosInsumosController.php
http://localhost:8000/api/insumos/	*get, post, put, delete*	insumosController.php
http://localhost:8000/api/tiposServicios/	*get, post, put, delete*	tiposServiciosController.php
http://localhost:8000/api/tiposEmergencias/	*get, post, put, delete*	tiposEmergenciasController.php
http://localhost:8000/api/servicios/	*get, post, put, delete*	serviciosController.php
http://localhost:8000/api/bomberosServicios/	*get, post, put, delete*	bomberosServiciosController.php
http://localhost:8000/api/unidadesServicios/	*get, post, put, delete*	unidadesServiciosController.php
http://localhost:8000/api/llamadasEmergencias/	*get, post, put, delete*	llamadasEmergenciasController.php
http://localhost:8000/api/bomberosEmergencias/	*get, post, put, delete*	bomberosEmergenciasController.php
http://localhost:8000/api/unidadesEmergencias	*get, post, put, delete*	unidadesEmergenciasController.php
http://localhost:8000/api/usuarios/	*get, post, put, delete*	usuariosController.php

Fuente: SASE29 (2021). *Rutas y controladores asociados.* Consultado el 10 de octubre de 2021. Recuperado de código fuente propio de sistema SASE29.

5.3. Páginas web

La aplicación web con la cual interactúan directamente los clientes desde su estación de trabajo esta desarrollada en Angular, se ha iniciado con el proceso de crear las diferentes pantallas de cada una de las funcionalidades del sistema, a continuación, se muestran algunas de las páginas realizadas utilizando el API como proveedor del contenido a mostrar.

5.3.1. Pantalla de inicio

La pantalla de inicio es la portada de la aplicación web, es la primera página que se cargará al dirigirse a la dirección https://cvb29compania.com, cuando la aplicación sea publicada en el dominio antes descrito, esto será lo primero que verán los visitantes del sitio.

Figura 40. **Página de portada 29 compañía de bomberos**

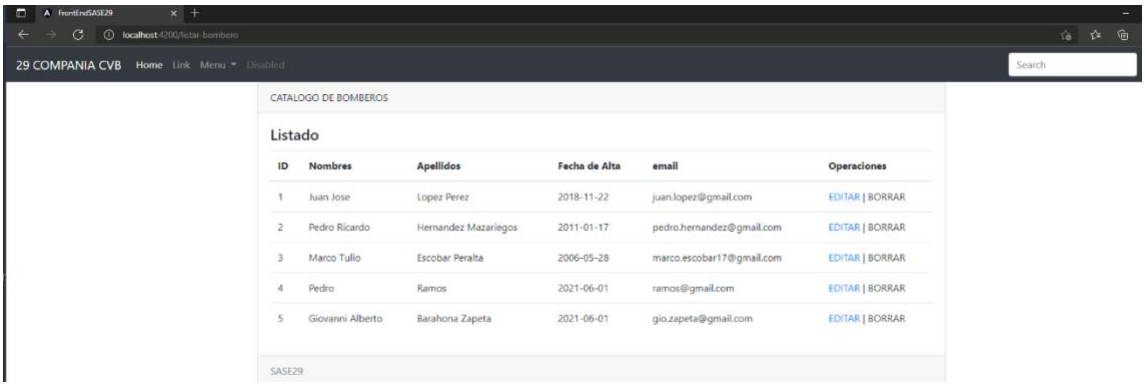
Fuente: elaboración propia, realizado con aplicación web local.

5.3.2. Prototipo página listar bomberos

Las páginas de las cuales se componen la aplicación consumirán el API para poder obtener los datos necesarios para mostrar en el contenido de la página, tal es el caso de la página que provee el listado de bomberos en el sistema, la página cargara el contenido consumiendo una ruta del controlador de bomberos, para así obtener el set de datos a mostrar.

En la siguiente figura podremos evidenciar la estructura de la página que consume el servicio que provee el API, en específico una petición HTTP *get* sobre la ruta de bomberos:

Figura 41. **Prototipo página de listado de bomberos**



Fuente: elaboración propia, realizado con aplicación web local.

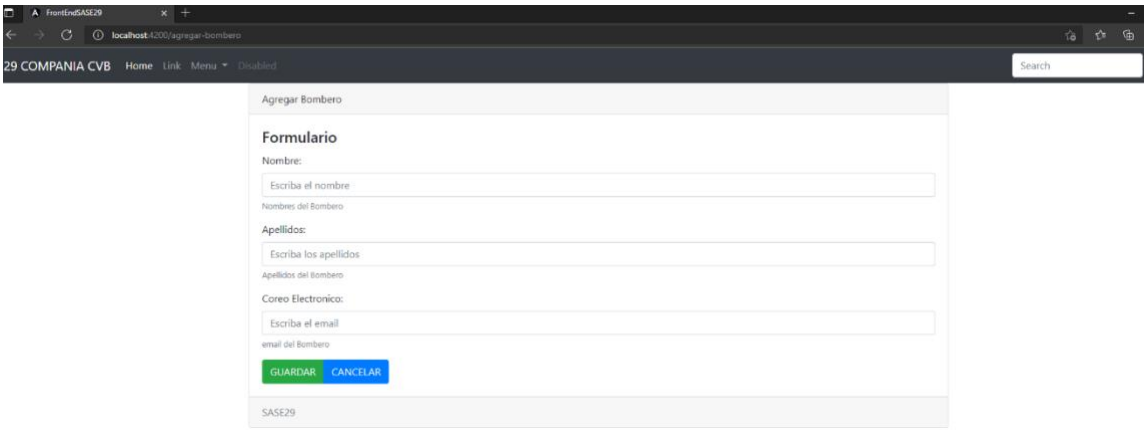
5.3.3. Prototipo página agregar bombero

En esta página, se podrá dar de alta a un nuevo bombero, para este cometido la página se conectará con la ruta del API concerniente a los bomberos

y procederá a consumir el controlador de bomberos con una petición HTTP *post* enviando los datos capturados en el formulario.

En la siguiente figura podemos apreciar el prototipo de esta página de captura de datos y los botones asociados para enviar dichos datos al API:

Figura 42. **Prototipo página agregar bombero**



Fuente: elaboración propia, realizado con aplicación web local.

5.4. Json web *token*

Dada su naturaleza, la aplicación web estará disponible desde internet hacia el mundo, por tal razón se debe asegurar que la aplicación estará protegida ante accesos no autorizados.

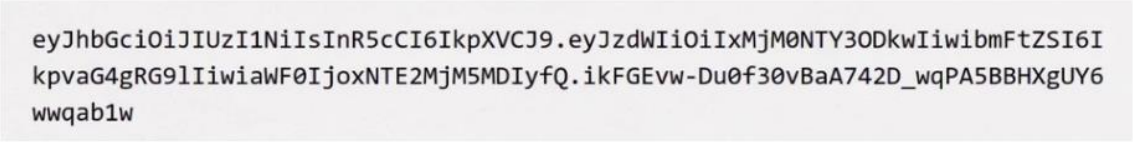
Con este fin, hemos aplicado un mecanismo de *TOKEN*, tanto del lado del *front end* como del *back end*, este mecanismo servirá para autenticar todas las peticiones desde la aplicación web hacia el API, mediante este mecanismo

implementaremos la autenticación de usuario y la validez de la autenticación en tiempo de sesión e identidad de quien lo emite.

JWT es un estándar de la industria, se encuentra definido en el documento RFC7519, el cual provee los mecanismos para poder propagar de manera segura la identidad de un usuario entre dos partes, además se podrá también indicar dentro de este mecanismo los privilegios con los que contará.

El llamado *TOKEN* el cual es generado a través de una petición de inicio de sesión con credenciales validas, es una cadena de texto, la cual está cifrada en Base64, la figura a continuación muestra la apariencia de un *TOKEN* generado.

Figura 43. **Json web *token***



Fuente: Open webinars (2020). *¿Qué es json web token y cómo funciona?* Consultado el 14 de noviembre de 2021. Recuperado de https://openwebinars.net/blog/que-es-json-web-token-y-como-funciona.

Cada *TOKEN* constara de tres partes cifradas, las cuales están estructuradas de la siguiente manera:

- *Header*: el encabezado en el cual se indicará el algoritmo en el cual está cifrado y el tipo de *token.*

- *Payload*: la carga registrada, contiene los datos del usuario y los privilegios con los que cuenta. También puede contener información personalizada.

- *Signature*: la firma, esta nos permitirá verificar la autenticidad del *TOKEN* emitido, por medio de esta firma vamos a verificar que el usuario sea quien dice ser y que la integridad del mensaje no se ha modificado por terceras partes.

Por medio de los 3 apartados anteriores, se podrá verificar que el *TOKEN* enviado sea válido, sea integro y contenga las credenciales de quien dice ser.

Como se mencionó anteriormente, cada uno de estos *TOKEN* tiene un ciclo de vida que se cumplirá en el tiempo que hayamos definido dentro de la *TOKEN factory*, la cual no es más que la entidad o instancia de *software* que se encargara de generar un nuevo *TOKEN* cuando sea requerido. El tiempo que un *TOKEN* sea válido será definido por el programador a petición del cliente o a criterio propio.

En la siguiente figura se muestra el ciclo de vida de un *TOKEN* generado, este ciclo se cumplirá a cabalidad mientras el *TOKEN* este vivo, es decir, aún sea válido, que no se encuentre expirado.

Figura 44. **Ciclo de vida de un json web *token***

Fuente: Open webinars (2020). *¿Qué es json web token y cómo funciona?.* Consultado el 14 de noviembre de 2021. Recuperado de https://openwebinars.net/blog/que-es-json-web-token-y-como-funciona.

5.5. Protegiendo el API

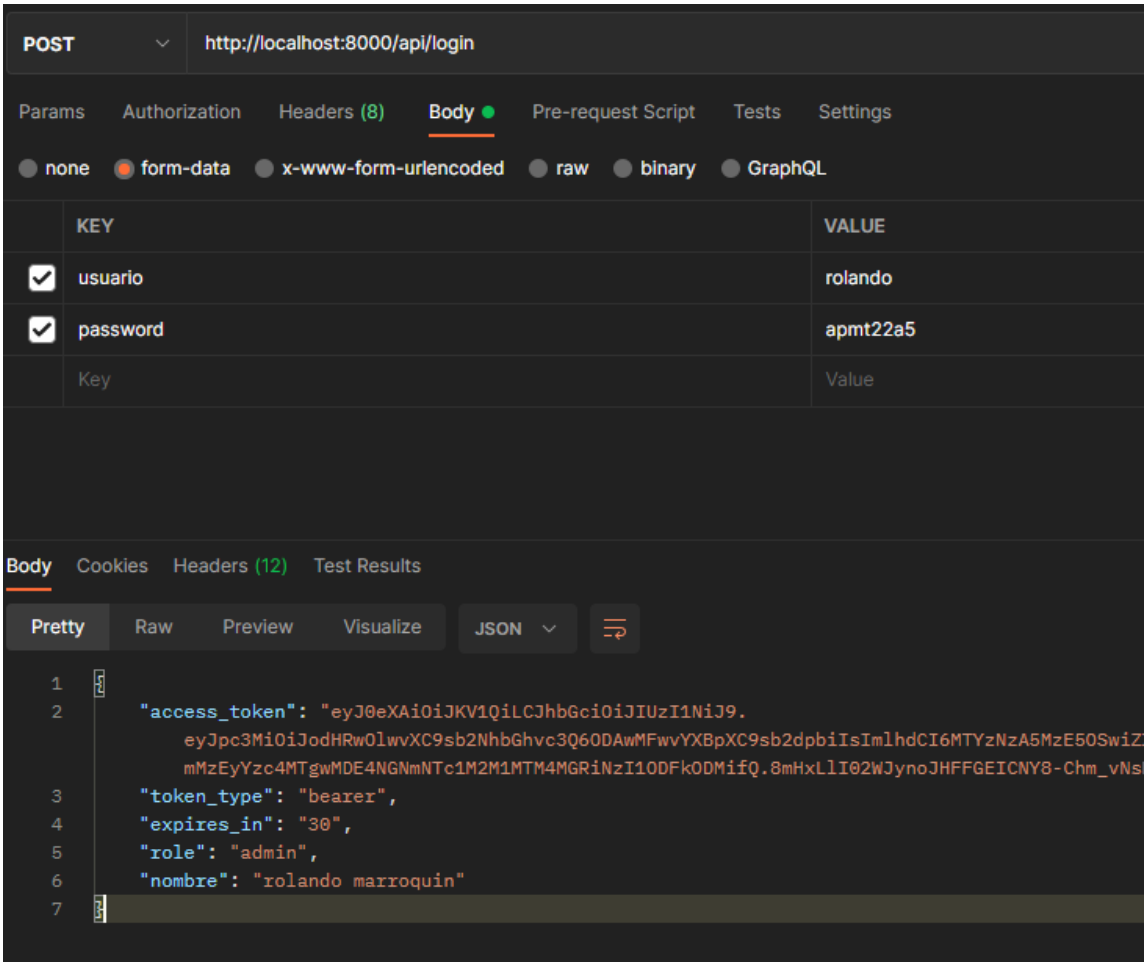
Cada una de las rutas expuestas por el API se encuentran protegidas mediante JWT, la única ruta que no lo estará será la de *login*, ya que esta ruta será la encargada de validar las credenciales del usuario y responder a estas, en caso las credenciales sean validas, con un *TOKEN* asociado a dicho usuario.

En cada ruta, el API atenderá la petición única y exclusivamente si se dan las siguientes condiciones:

- La petición HTTP va acompañada de un *TOKEN*, en caso contrario se rechaza la petición.
- El *TOKEN* enviado en la petición HTTP es válido y no esta expirado.

En la siguiente imagen se podrá apreciar el resultado de un *login* satisfactorio, en el cual se responderá con información del usuario autenticado y el *TOKEN* asociado.

Figura 45. **Petición HTTP *get* a ruta *login***



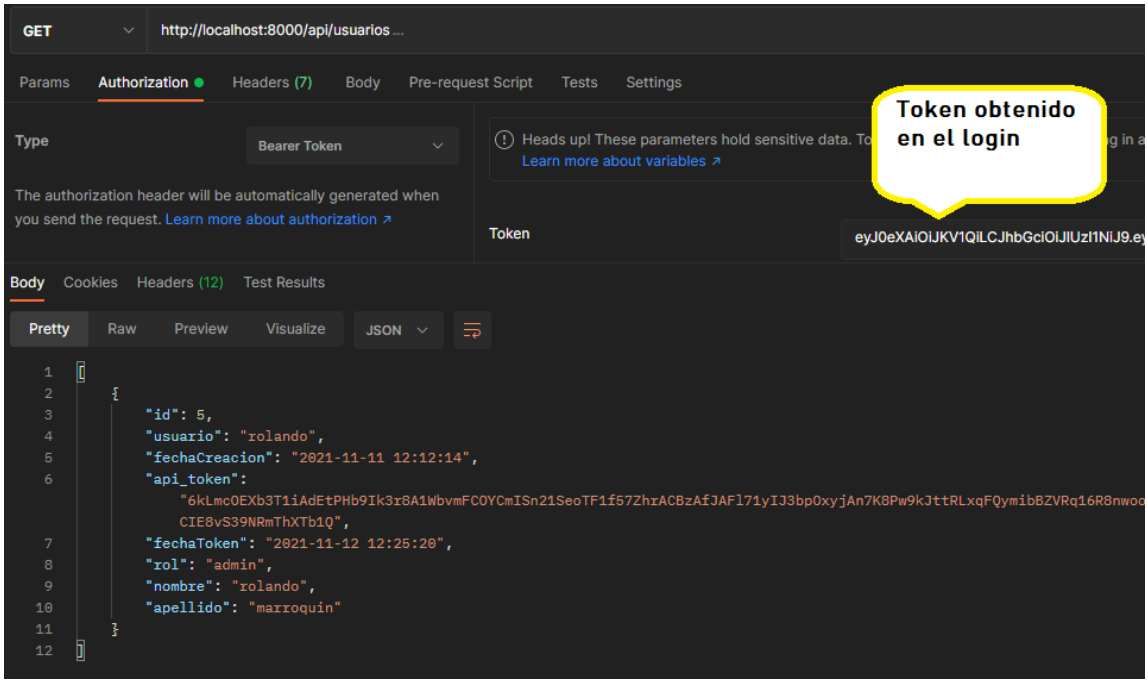
Fuente: elaboración propia, realizado con Postman.

Como podemos apreciar en la figura anterior, dentro de la respuesta a una petición de *login* el API devuelve la propiedad *ACCESS_TOKEN* la cual contiene el *TOKEN* asignado al usuario autenticado. Se puede apreciar que en la

respuesta también se indica el rol asociado al usuario, que para este caso es administrador, además se indica el tiempo de vida del *TOKEN,* el cual para efectos de pruebas tiene un tiempo asignado de 30 minutos.

Para cada ruta de petición de datos en el API se deberá adjuntar en el encabezado de seguridad el valor del *TOKEN* para que la petición sea atendida. En caso contrario se responderá con un mensaje de error que indicará que el usuario no está autorizado y se indicará el código de error 401.

Figura 46. **Petición HTTP *get* con *token* de autenticación**

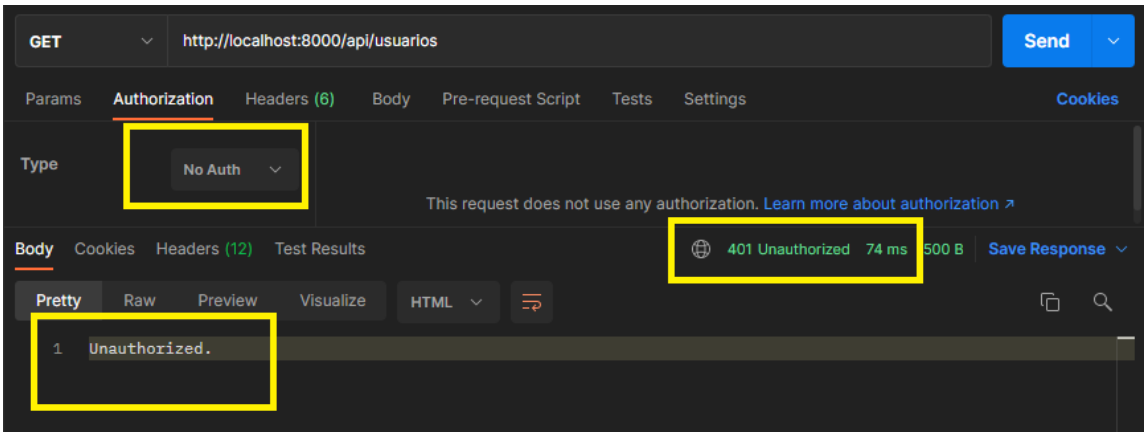


Fuente: elaboración propia, realizado con Postman.

En el caso que no se provea de un *TOKEN* de autenticación a una ruta protegida, el API procederá a rechazar la petición e indicar que no se tiene

autorización con un HTTP *STATUS CODE* 401, la siguiente figura muestra evidencia de ello.

Figura 47. **Petición HTTP *get* sin *token* de autorización**



Fuente: elaboración propia, realizado con Postman.

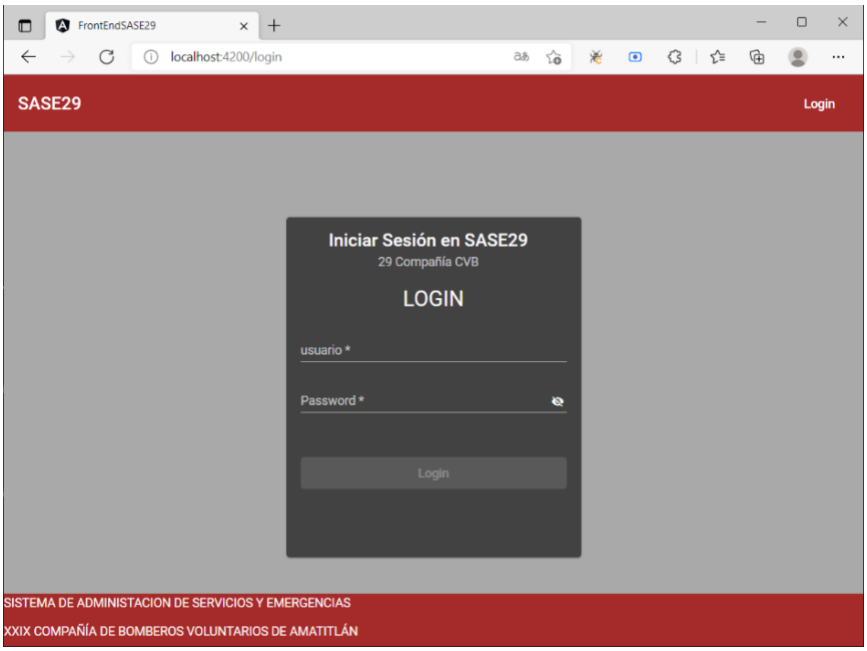
5.6. Protegiendo aplicación web

Dado que nuestra API ya se encuentra protegida con el estándar *JWT* (Json Web *TOKEN)* debemos entonces proveer de un mecanismo de autenticación en la aplicación web, para esto Angular provee herramientas que permiten validar obtener el *TOKEN* generado en la autenticación del usuario y utilizarlo en las peticiones posteriores.

Se ha desarrollado una página de inicio de sesión (comúnmente conocido como *Login*) en la cual el usuario se encontrará con un formulario reactivo de Angular Material, este formulario recopilará la información relacionada a las credenciales de un usuario que quiera iniciar sesión, consumiendo la ruta de *Login* de nuestro API.

En la siguiente figura se observa el componente de *Login* de la aplicación web.

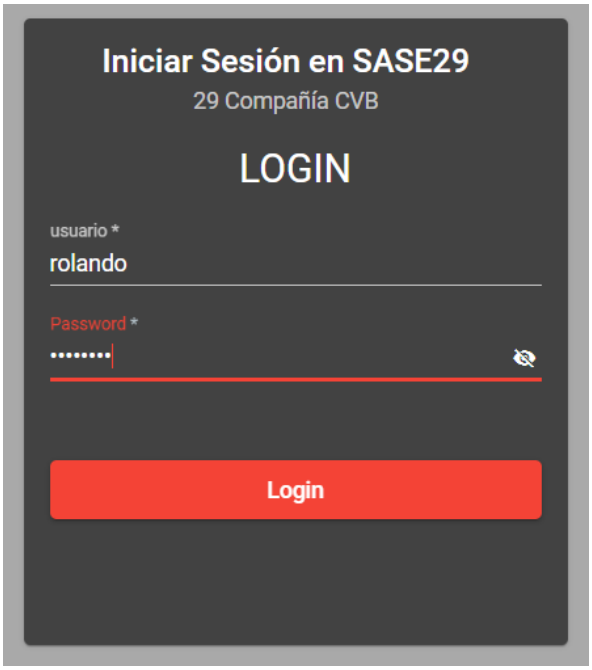
Figura 48. **Componente *login* web app**



Fuente: elaboración propia, realizado con aplicación web local.

El formulario de la figura es reactivo, es decir, que validara que los campos de usuario y *password* estén con un valor para poder activar el botón de *login.*

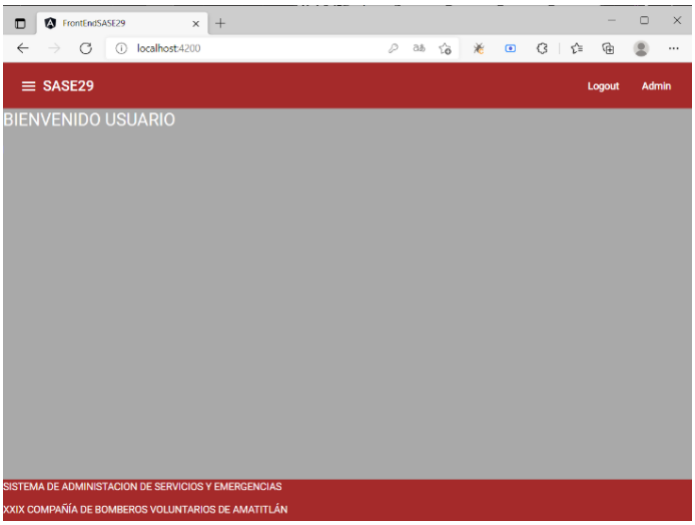
Figura 49. **Formulario reactivo *login***



Fuente: elaboración propia, realizado con aplicación web local.

Al momento de presionar el botón de *login* la aplicación se comunicará con el API para realizar la autenticación de las credenciales indicadas en el formulario, si esta autenticación resulta exitosa, la aplicación procederá a iniciar sesión con el usuario indicado y almacenar el *TOKEN* devuelto por el API en conjunto con el rol asignado al usuario.

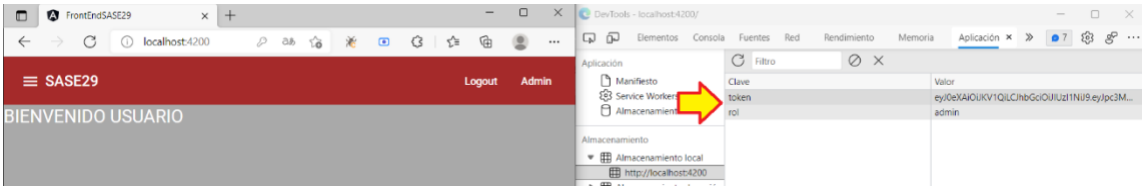
Figura 50. ***Login* exitoso**



Fuente: elaboración propia, realizado con aplicación web local.

Como podemos evidenciar, al momento de autenticar al usuario de manera exitosa, se dejará acceder al usuario a la aplicación y se activará el menú de opciones en la parte superior izquierda, este será un menú colapsable con las diferentes opciones.

Figura 51. ***Token* almacenado en la aplicación**

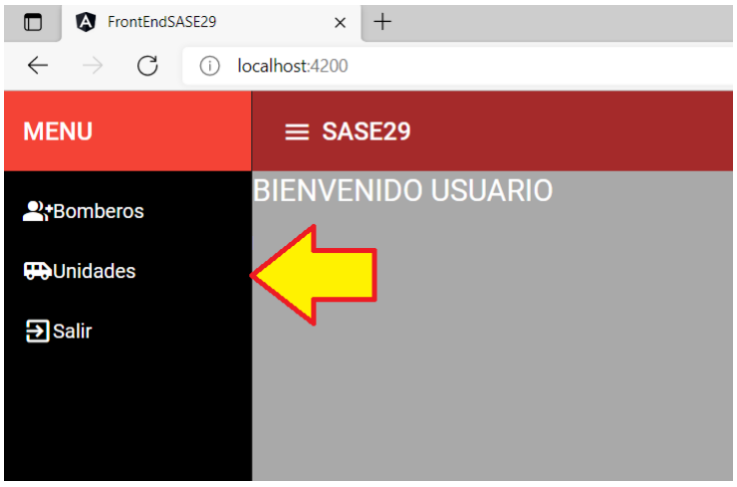


Fuente: elaboración propia, realizado con aplicación web local.

Como podemos apreciar en la figura anterior, la aplicación guarda en su almacenamiento local el valor del *TOKEN* obtenido durante el *Login*, además se guarda el rol asignado para poder mostrar opciones exclusivas del rol que se haya asignado al usuario, en este caso el rol obtenido es *Admin*, el cual corresponde a un usuario administrador. Como se mencionó anteriormente, al momento de iniciar sesión se dará acceso al menú colapsable de opciones, en las cuales se encontrarán los catálogos y operaciones del sistema.

En la siguiente figura se aprecia el menú de opciones, el cual es visible únicamente cuando se inicia sesión en el sistema con un *TOKEN* válido y vigente.

Figura 52. **Menú colapsable de opciones**

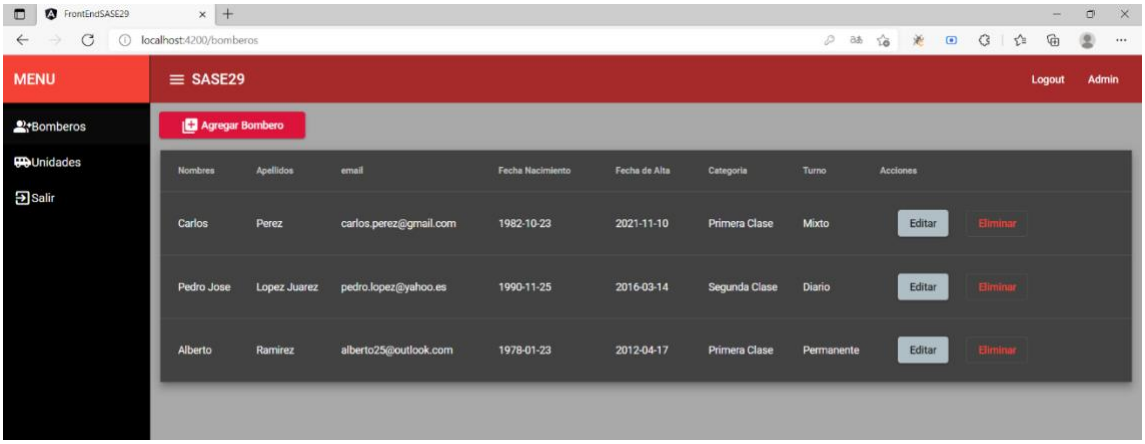


Fuente: elaboración propia, realizado con aplicación web local.

Desde este menú colapsable, el usuario podrá dirigirse a las pantallas de operaciones o catálogos, en cualquier momento a comodidad del usuario, este menú se podrá ocultar (colapsar) o mostrar para así tener una interfaz más amigable y adaptativa según sea el usuario.

Cada una de las diferentes pantallas de las cuales está compuesta la aplicación web, estará protegida contra un usuario no autenticado, ya que cada una en su inicio pedirá la autorización al componente de autorizaciones para saber si debe presentar la pantalla a un usuario autenticado. Por ejemplo, la opción de bomberos del menú colapsable mostrara en su inicio un listado de bomberos que se encuentren registrados. Esto siempre y cuando el *TOKEN* almacenado por la aplicación corresponda un usuario autenticado y que su sesión no este expirada.

Figura 53. **Página de bomberos registrados**

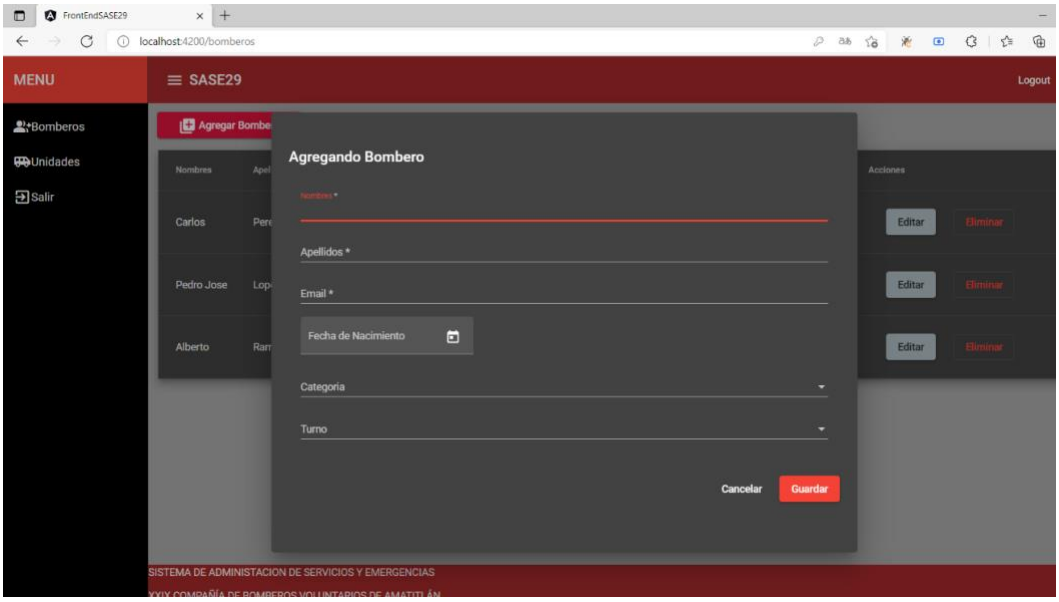


Fuente: elaboración propia, realizado con aplicación web local.

En la figura anterior podemos ver que se mostró el listado de bomberos registrados en el sistema porque el *TOKEN* que se utilizó para enviar la petición HTTP *get* de bomberos es válido y esté vigente aun, en el caso que no se tuviera un *TOKEN* valido o que este ya estuviera expirado, se redirige inmediatamente al usuario hacia la pantalla de *login* para que realice nuevamente la autenticación.

El mismo caso ocurrirá cuando se quiera agregar un nuevo bombero, se verificará si el usuario esta autenticado y la sesión es válida.

Figura 54. **Formulario para agregar bomberos**



Fuente: elaboración propia, realizado con aplicación web local.

El formulario anterior también es reactivo, es decir, que validara que cada campo ingresado contenga un valor valido para poder enviar la petición de agregación del bombero, además enviara de igual manera el *TOKEN* de autenticación hacia el API para que se acepte la petición HTTP y el nuevo bombero quede registrado.

5.7. Sistema publicado

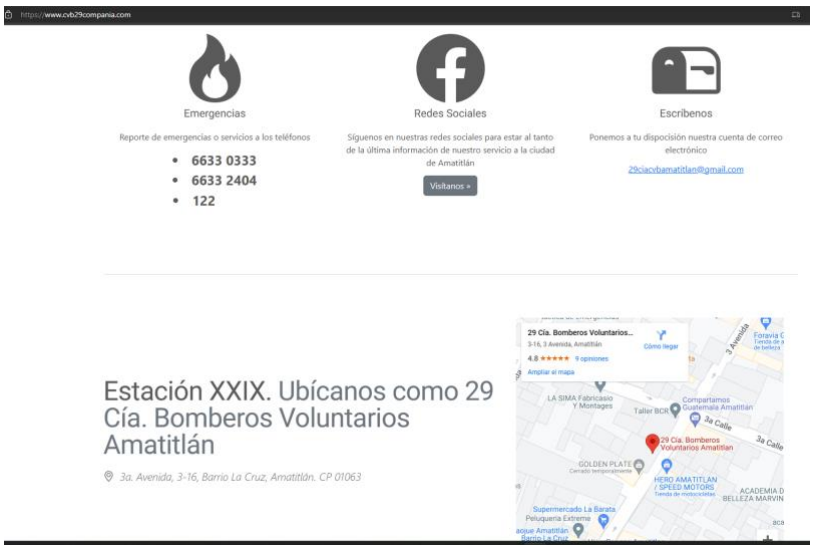
Como parte final de este trabajo de graduación, se ha publicado el sistema completo en el hosting adquirido para tal cometido. Vamos a listar las partes de

cada página, así como una descripción del cometido de cada una y evidenciar su funcionamiento.

5.7.1. *Landing page*

Esta página es la página que se carga inicialmente al acceder a la dirección https://www.cvb29compania.com/, en ella se muestra la portada del sistema, información de contacto, autoridades, ubicación, entre otros.

Figura 55. ***Landing page* 29 compañía de Bomberos Voluntarios**

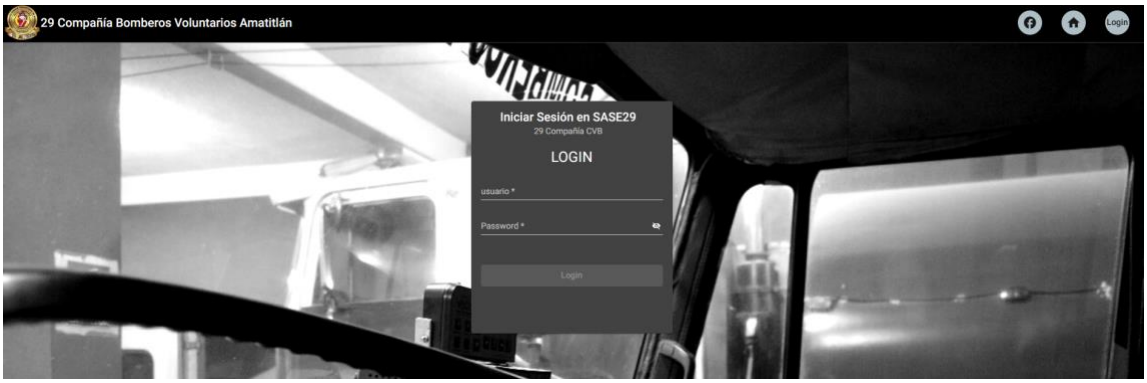


Fuente: elaboración propia, realizado con aplicación web local.

5.7.2. *Login*

Esta página tiene como objetivo el autenticar a los usuarios que quieran ingresar al sistema SASE29, se deberá proveer un usuario y contraseña para poder ingresar.

Figura 56. ***Login page* 29 compañía de Bomberos Voluntarios**

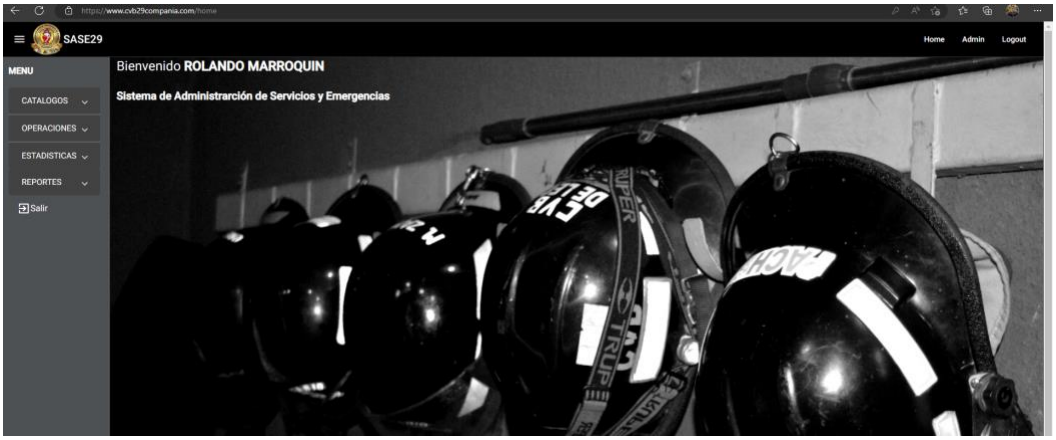


Fuente: elaboración propia, realizado con aplicación web local.

5.7.3. *Home*

Esta página se mostrará cuando el usuario se haya autenticado de manera correcta, se le dará la bienvenida el sistema y se cargaran las opciones de menú que estén asociadas al rol del usuario autenticado.

Figura 57. **Página de inicio del sistema SASE29**



Fuente: elaboración propia, realizado con aplicación web local.

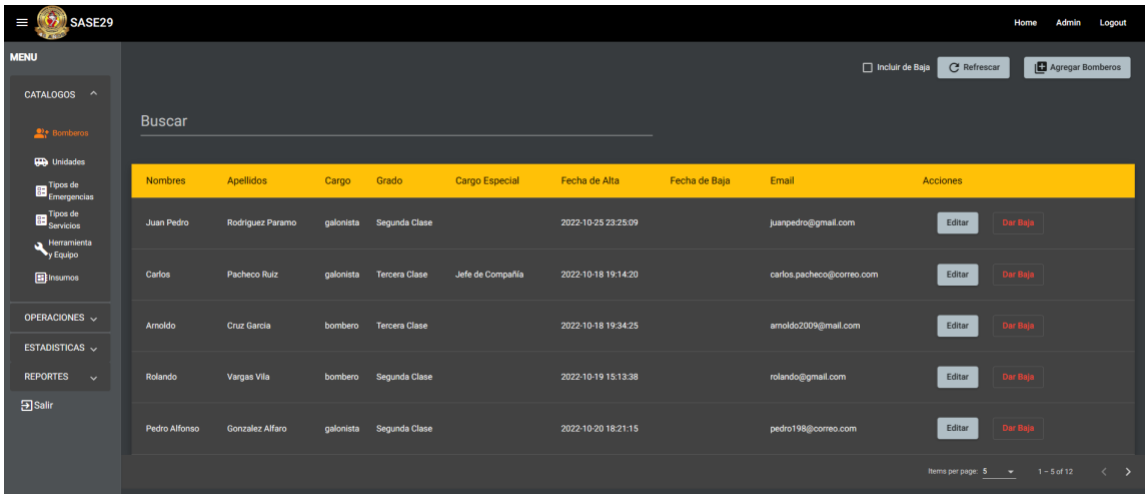
5.7.4. Catálogos

Estas páginas tienen como principal función el poder captar toda la información que será necesaria para la operación del sistema en el día a día, se podrá ingresar información de bomberos, unidades, herramientas, insumos, entre otros.

5.7.4.1. Catálogo de bomberos

Pagina para captar la información de bomberos existentes, dar de alta a nuevos bomberos y dar de baja.

Figura 58. **Catálogo de bomberos sistema SASE29**

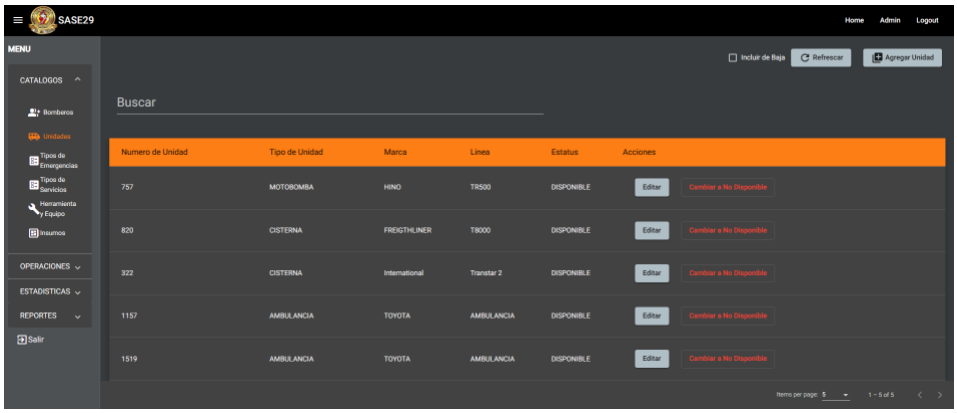


Fuente: elaboración propia, realizado con aplicación web local.

5.7.4.2. Catálogo de unidades

En esta página se podrán crear nuevas unidades, administrar las unidades existentes, las cuales estarán disponibles para poder ser asignadas a llamadas de emergencia y servicios.

Figura 59. **Catálogo de unidades sistema SASE29**

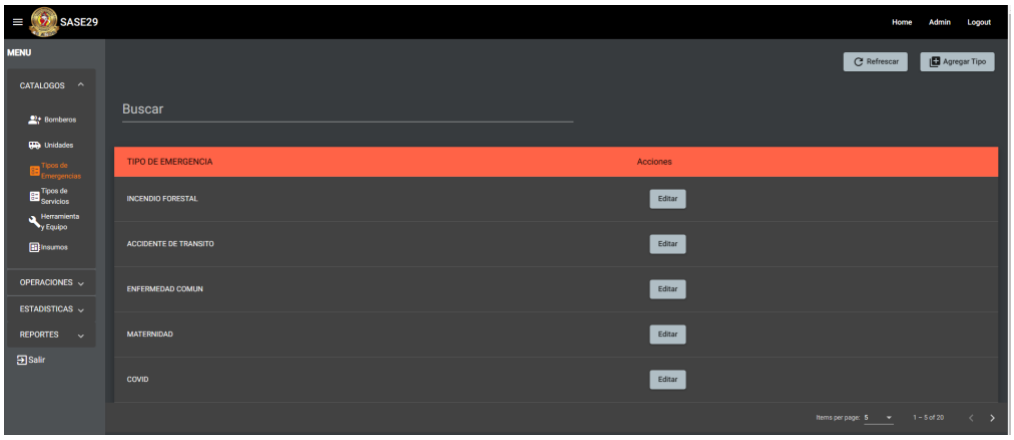


Fuente: elaboración propia, realizado con aplicación web local.

5.7.4.3. Tipos de emergencias

Esta página tiene como cometido el administrar los tipos de emergencias que estarán disponibles en el sistema. Se podrán crear nuevas emergencias, editar las ya existentes.

Figura 60. **Catálogo de tipos de emergencias sistema SASE29**

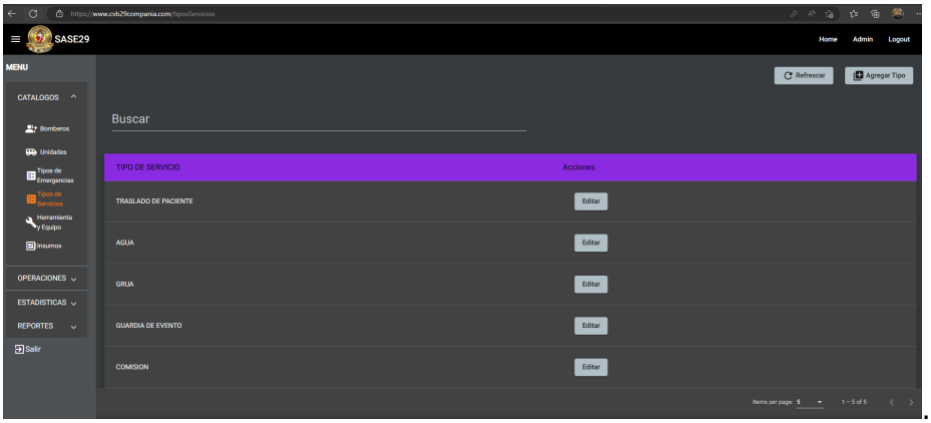


Fuente: elaboración propia, realizado con aplicación web local.

5.7.4.4. Tipos de servicios

En esta página se podrán administrar los diferentes tipos de servicios que estarán disponibles en el sistema, se podrá agregar nuevos y editar los ya existentes.

Figura 61. **Catálogo de tipos de servicios sistema SASE29**

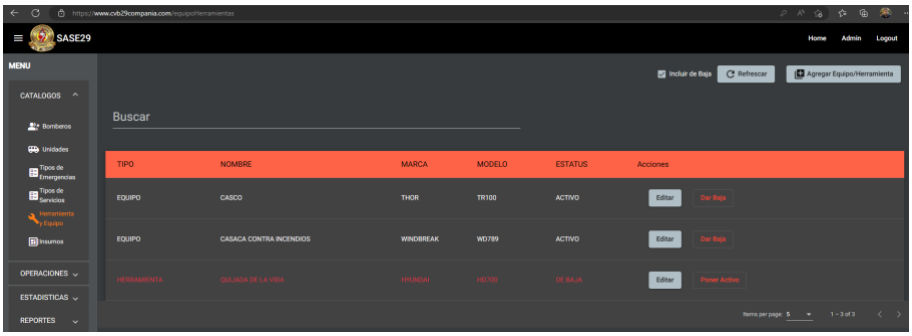


Fuente: elaboración propia, realizado con aplicación web local.

5.7.4.5. Catálogo de herramienta y equipo

Esta página proveerá los controles para poder administrar el listado de herramientas y equipo disponible en el sistema. Al momento de ingresar un nuevo elemento se deberá especificar si se trata de una herramienta o un equipo.

Figura 62. **Catálogo de herramientas y equipo sistema SASE29**



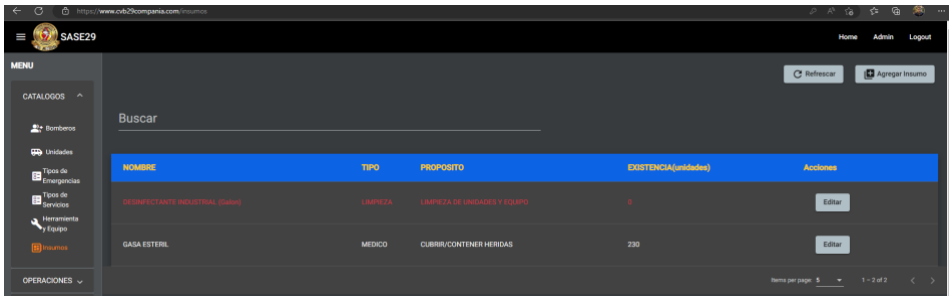
Fuente: elaboración propia, realizado con aplicación web local.

5.7.4.6. Catálogo de insumos

En este catálogo, la página mostrara el listado de insumos que están disponibles en el sistema, se deberá indicar el tipo de insumo y su propósito, además de llevar un control de las existencias de este.

Se proveen también los controles necesarios para poder editar toda la información relacionada al insumo ingresado.

Figura 63. **Catálogo de herramientas y equipo sistema SASE29**



Fuente: elaboración propia, realizado con aplicación web local.

5.7.5. Operaciones

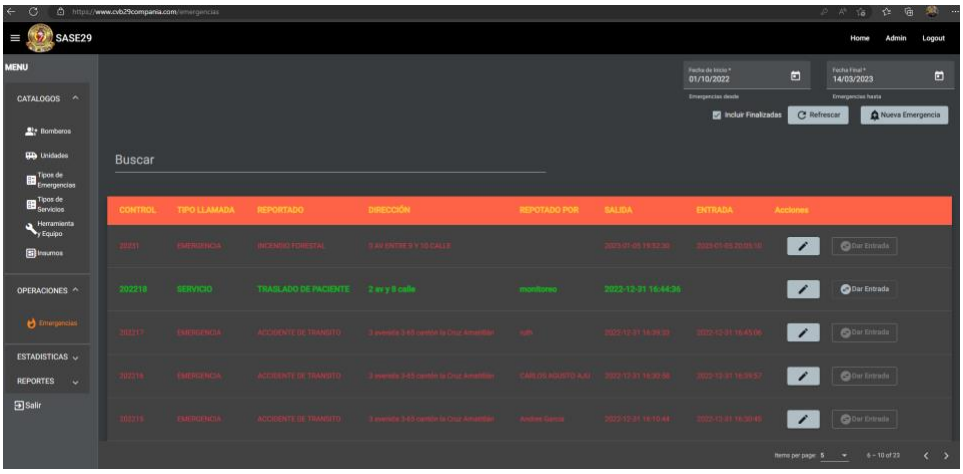
Las operaciones del sistema se basan en el registro de las llamadas de emergencias y servicios que se atienden día a día en la estación de bomberos. Estas operaciones estarán alimentadas por los datos que previamente se definan en los catálogos del sistema.

5.7.5.1. Registro de llamadas

En esta página se deberán registrar las llamadas que entran a la cabina de bomberos solicitando ya sea un servicio o la atención de una emergencia.

Esta página servirá también para poder visualizar las emergencias o servicios del día que se estén atendiendo, cabe resaltar que las llamadas que se estén atendiendo resaltaran en color verde para indicar que están en atención, las llamadas registradas pero que un no están en atención se verán con el color de la fuente normal (blanco), para las llamadas que ya fueron atendidas se mostraran en color rojo.

Figura 64. **Pantalla operativa de control de llamadas sistema SASE29**



Fuente: elaboración propia, realizado con aplicación web local.

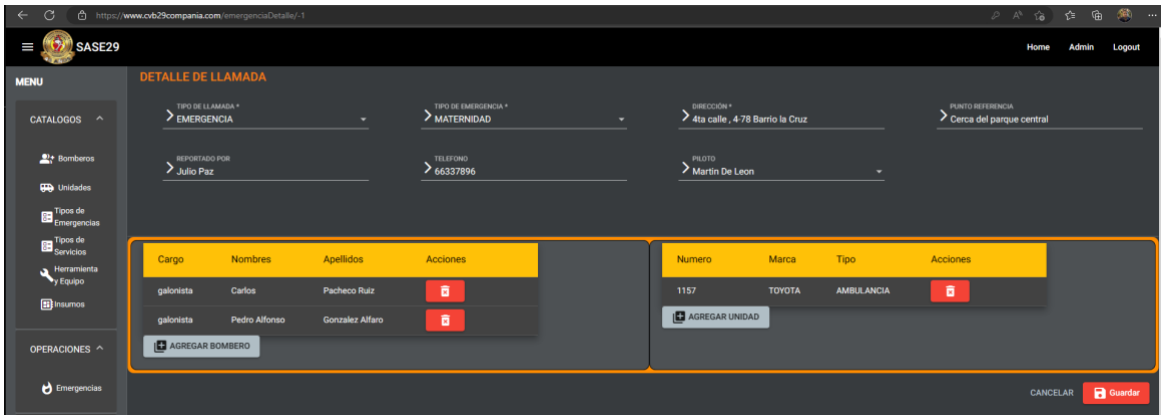
Para registrar una nueva llamada se deberá hacer clic sobre el botón Nueva Emergencia, esta acción nos llevará a la página en la cual podremos registrar los datos de la llamada de emergencia o servicio.

Como primer paso se deberá seleccionar si se trata de una llamada de emergencia o servicios, posteriormente se deberá seleccionar el tipo de llamada. Acto seguido debemos de ingresar la dirección de donde se requiere la unidad, quien lo reporta, el teléfono de donde se reporta, el piloto designado.

Esta pantalla también proveerá controles adicionales para poder indicar los bomberos que atenderán la emergencia, así como también la o las unidades destacadas para la atención de la llamada.

En la siguiente figura podemos evidenciar los controles mencionados previamente.

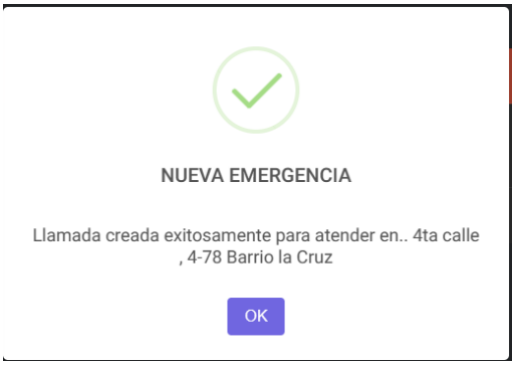
Figura 65. **Pantalla operativa de control de llamadas sistema SASE29**



Fuente: elaboración propia, realizado con aplicación web local.

Cuando la creación de la llamada fue exitosa se mostrará la siguiente alerta de confirmación:

Figura 66. **Alerta de confirmación exitosa sistema SASE29**



Fuente: elaboración propia, realizado con aplicación web local.

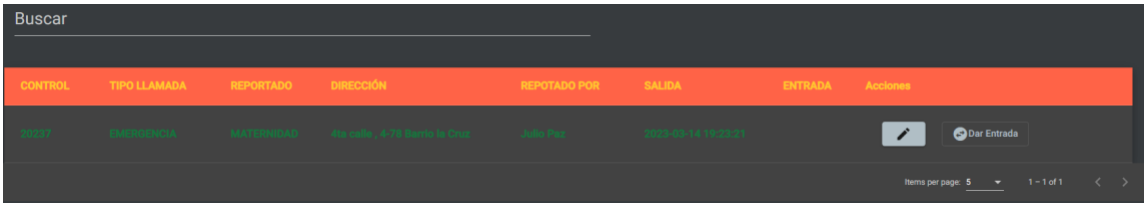
En este momento la nueva llamada será visible en la página de control de llamadas de emergencia, en la cual podremos darle salida cuando la unidad de bomberos esté lista.

5.7.5.2. Salidas y entradas de emergencias

Para cada una de las llamadas registradas en el sistema se deberá indicar el momento de salida de una unidad motorizada para la atención de la emergencia, de igual manera se indicará cuando la unidad finalice la atención de la emergencia regresando a la base.

Para este cometido se provee de un botón en el cual, cual la emergencia aun no haya salido a atención, el botón dirá Dar Salida, en caso contrario cuando la emergencia este en atención y se requiera finalizarla el botón dirá Dar Entrada, el sistema procederá a guardar las fechas y horas de salida y entrada de la unidad para el control y reportería necesaria.

Figura 67. **Llamada en atención en sistema SASE29**



Fuente: elaboración propia, realizado con aplicación web local.

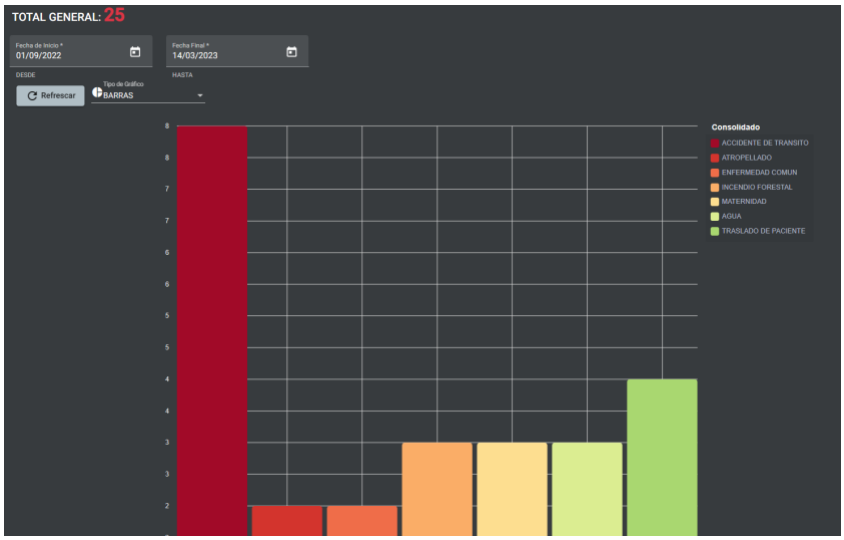
Figura 68. **Llamada finalizada en sistema SASE29**

Fuente: elaboración propia, realizado con aplicación web local.

5.7.6. Estadísticas

Es de vital importancia tener de manera instantánea las estadísticas de llamadas atendidas durante el día o un periodo de tiempo determinado. Para tal cometido el sistema provee de un módulo de estadísticas interactivas en el cual se podrá visualizar de manera gráfica las estadísticas del sistema.

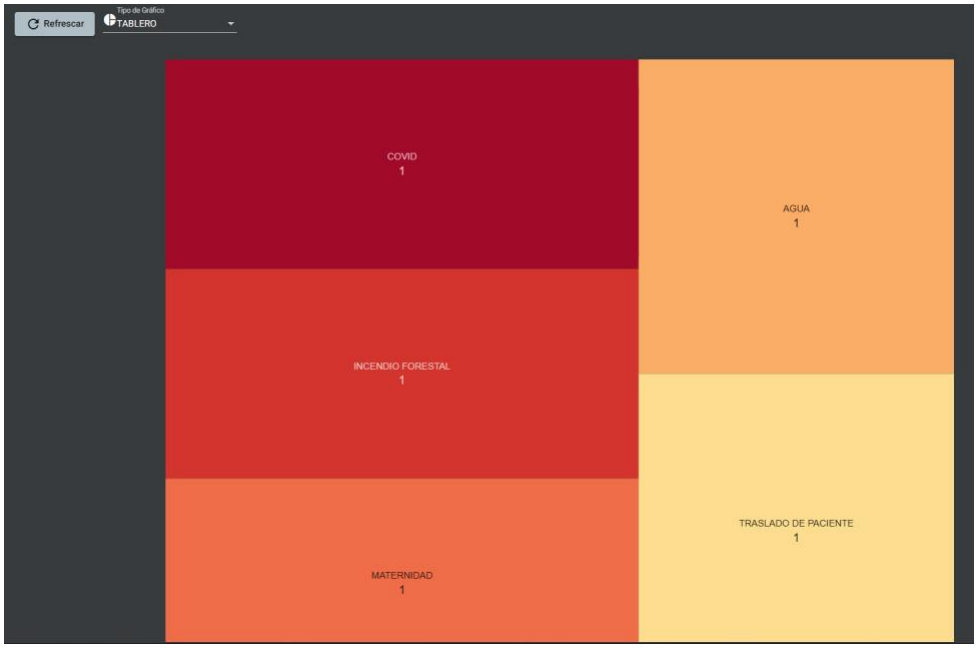
Figura 69. **Estadísticas consolidadas de llamadas atendidas**



Fuente: elaboración propia, realizado con aplicación web local.

Las estadísticas se pueden obtener consolidadas o únicamente de servicios o emergencias, según sea la necesidad. También se pueden visualizar mediante un gráfico de barras, un gráfico de pie o un tablero interactivo que nos dará la información en casillas.

Figura 70. **Estadísticas de llamadas en tablero interactivo**



Fuente: elaboración propia, realizado con aplicación web local.

5.7.7. Reportes

La reportería es de vital importancia para la compañía, ya que este será el reflejo palpable de las operaciones realizadas en el sistema durante un tiempo determinado.

El sistema proveerá un menú de reportes en el cual el usuario podrá seleccionar los siguientes reportes:

- Reporte de emergencias y servicios
- Reporte de personal
- Reporte de insumos
- Reporte de herramienta y equipo

Figura 71. **Reporte de emergencias y servicios en sistema SASE29**

REPORTE DE EMERGENCIAS Y SERVICIOS
29 Compañía de Bomberos Voluntarios Amatitlán
Del 14-03-2023 al 14-03-2023
Generado por ROLANDO MARROQUIN , el día 14-03-2023

CONTROL	TIPO	ATENDIDO	SALIDA	ENTRADA	DURACION (Minutos)
202311	EMERGENCIA	COVID	2023-03-14 19:46:59	2023-03-14 19:52:12	5.22
202310	SERVICIO	AGUA	2023-03-14 19:47:03		NaN
20239	SERVICIO	TRASLADO DE PACIENTE			NaN
20238	EMERGENCIA	INCENDIO FORESTAL			NaN
20237	EMERGENCIA	MATERNIDAD	2023-03-14 19:23:21	2023-03-14 19:29:06	5.75

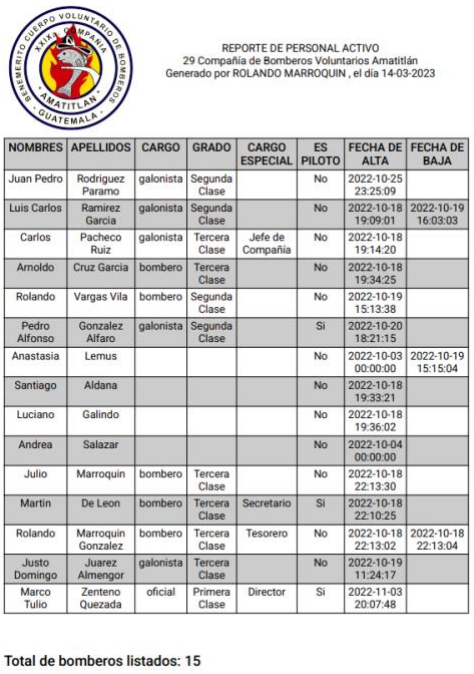
RESUMEN

TIPO	CANTIDAD
COVID	1
INCENDIO FORESTAL	1
MATERNIDAD	1
AGUA	1
TRASLADO DE PACIENTE	1

TOTAL: 5 Servicios atendidos.

Fuente: elaboración propia, realizado con aplicación web local.

Figura 72. **Reporte de personal en sistema SASE29**



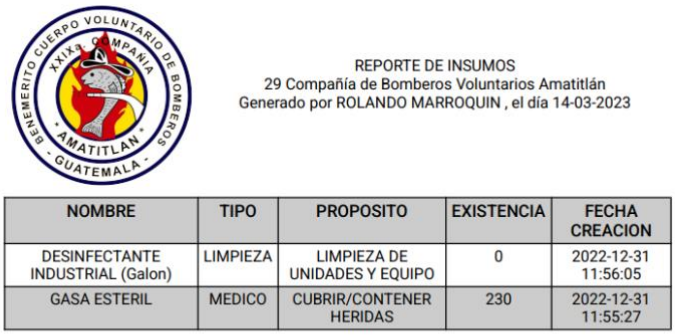
REPORTE DE PERSONAL ACTIVO
29 Compañía de Bomberos Voluntarios Amatitlán
Generado por ROLANDO MARROQUIN , el día 14-03-2023

NOMBRES	APELLIDOS	CARGO	GRADO	CARGO ESPECIAL	ES PILOTO	FECHA DE ALTA	FECHA DE BAJA
Juan Pedro	Rodriguez Paramo	galonista	Segunda Clase		No	2022-10-25 23:25:09	
Luis Carlos	Ramirez Garcia	galonista	Segunda Clase		No	2022-10-18 19:09:01	2022-10-19 16:03:03
Carlos	Pacheco Ruiz	galonista	Tercera Clase	Jefe de Compañia	No	2022-10-18 19:14:20	
Arnoldo	Cruz Garcia	bombero	Tercera Clase		No	2022-10-18 19:34:25	
Rolando	Vargas Vila	bombero	Segunda Clase		No	2022-10-19 15:13:38	
Pedro Alfonso	Gonzalez Alfaro	galonista	Segunda Clase		Si	2022-10-20 18:21:15	
Anastasia	Lemus				No	2022-10-03 00:00:00	2022-10-19 15:15:04
Santiago	Aldana				No	2022-10-18 19:33:21	
Luciano	Galindo				No	2022-10-18 19:36:02	
Andrea	Salazar				No	2022-10-04 00:00:00	
Julio	Marroquin	bombero	Tercera Clase		No	2022-10-18 22:13:30	
Martin	De Leon	bombero	Tercera Clase	Secretario	Si	2022-10-18 22:10:25	
Rolando	Marroquin Gonzalez	bombero	Tercera Clase	Tesorero	No	2022-10-18 22:13:02	2022-10-18 22:13:04
Justo Domingo	Juarez Almengor	galonista	Tercera Clase		No	2022-10-19 11:24:17	
Marco Tulio	Zenteno Quezada	oficial	Primera Clase	Director	Si	2022-11-03 20:07:48	

Total de bomberos listados: 15

Fuente: elaboración propia, realizado con aplicación web local.

Figura 73. **Reporte de insumos en sistema SASE29**



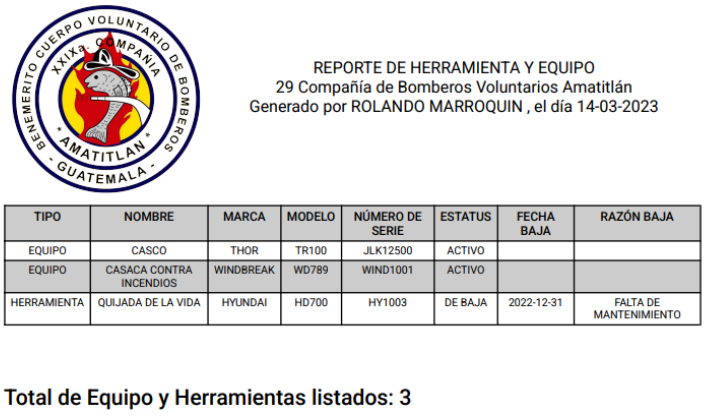
REPORTE DE INSUMOS
29 Compañía de Bomberos Voluntarios Amatitlán
Generado por ROLANDO MARROQUIN , el día 14-03-2023

NOMBRE	TIPO	PROPOSITO	EXISTENCIA	FECHA CREACION
DESINFECTANTE INDUSTRIAL (Galon)	LIMPIEZA	LIMPIEZA DE UNIDADES Y EQUIPO	0	2022-12-31 11:56:05
GASA ESTERIL	MEDICO	CUBRIR/CONTENER HERIDAS	230	2022-12-31 11:55:27

Total de insumos listados: 2

Fuente: elaboración propia, realizado con aplicación web local.

Figura 74. **Reporte de herramienta y equipo en sistema SASE29**



REPORTE DE HERRAMIENTA Y EQUIPO
29 Compañía de Bomberos Voluntarios Amatitlán
Generado por ROLANDO MARROQUIN , el día 14-03-2023

TIPO	NOMBRE	MARCA	MODELO	NÚMERO DE SERIE	ESTATUS	FECHA BAJA	RAZÓN BAJA
EQUIPO	CASCO	THOR	TR100	JLK12500	ACTIVO		
EQUIPO	CASACA CONTRA INCENDIOS	WINDBREAK	WD789	WIND1001	ACTIVO		
HERRAMIENTA	QUIJADA DE LA VIDA	HYUNDAI	HD700	HY1003	DE BAJA	2022-12-31	FALTA DE MANTENIMIENTO

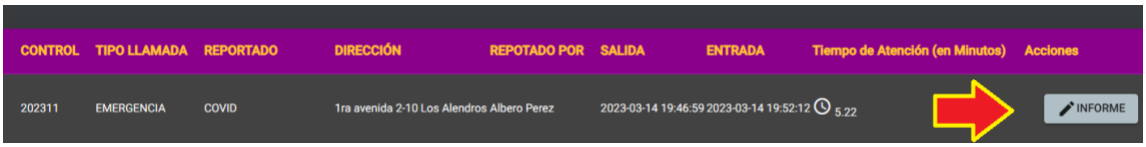
Total de Equipo y Herramientas listados: 3

Fuente: elaboración propia, realizado con aplicación web local.

5.7.8. Informe de llamada

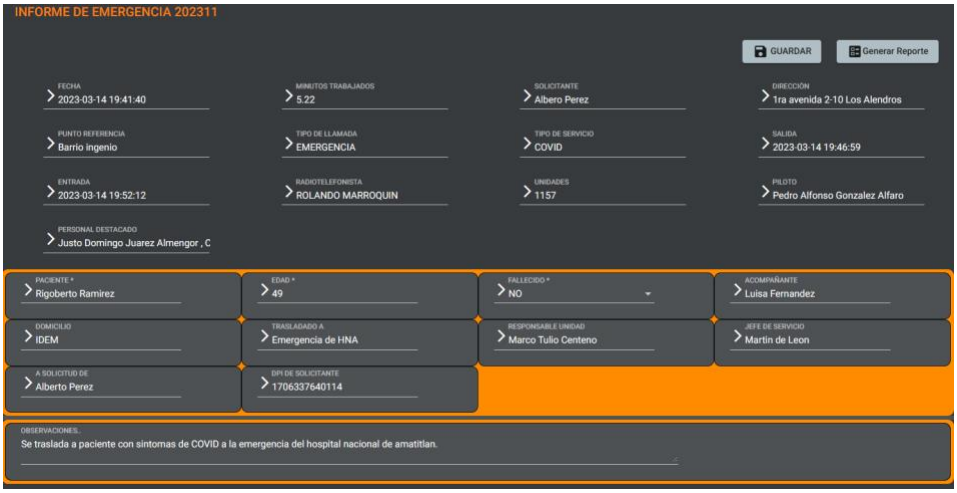
Existen casos en donde para las llamadas de emergencias y servicios atendidos, ya sea por parte del afectado o autoridades competentes, se requiere un informe con detalles de la atención prestada por parte de la compañía de bomberos. Para este cometido se debe realizar un informe de la llamada atendida, este informe se debe generar desde la pantalla de reporte de llamadas atendidas, cada llamada listada tendrá asociado un informe, el cual se deberá de llenar con los detalles de lo realizado durante la atención y los datos del solicitante.

Figura 75. **Botón para generar informe de llamada en sistema SASE29**



Fuente: elaboración propia, realizado con aplicación web local.

Figura 76. **Pantalla para generar informe de llamada en sistema SASE29**



Fuente: elaboración propia, realizado con aplicación web local.

Figura 77. **Informe generado de llamada en sistema SASE29**

INFORME DE LLAMADA DE EMERGENCIA
29 Compañía de Bomberos Voluntarios Amatitlán
Generado por ROLANDO MARROQUIN , el día 14-03-2023

CONTROL: 202311 FECHA: 14-03-2023 19:41:40 MINUTOS TRABAJADOS: 5.22

TELÉFONO: 66337845 SOLICITANTE: Alberto Perez

DIRECCIÓN: 1ra avenida 2-10 Los Alendros

PUNTO DE REFERENCIA: Barrio ingenio

TIPO DE LLAMADA: EMERGENCIA CLASE DE SERVICIO: COVID

SALIDA DE: 29a. Cía. CVB HORA: 19:46:59 ENTRADA A: 29a. Cía. CVB HORA: 19:52:12

RADIOTELEFONISTA: ROLANDO MARROQUIN UNIDADES: 1157

PILOTO(S): Pedro Alfonso Gonzalez Alfaro

NOMBRE DEL (LOS) PACIENTE(S): Rigoberto Ramirez

EDAD(ES): 49 FALLECIDO(S): NO ACOMPAÑANTE: Luisa Fernandez

DOMICILIO: IDEM

TRASLADADO A: Emergencia de HNA

PERSONAL DESTACADO: Justo Domingo Juarez Almengor , Carlos Pacheco Ruiz

OBSERVACIONES:
Se traslada a paciente con sintomas de COVID a la emergencia del hospital nacional de amatitlan.

RESPONSABLE DE UNIDAD: Marco Tulio Centeno FIRMA: _____

ES CONFORME AL PILOTO: Pedro Alfonso Gonzalez Alfaro FIRMA: _____

Vo.Bo. JEFE DE SERVICIO: Martin de Leon FIRMA: _____

SE EXTIENDE COPIA CERTIFICADA DE ESTE REPORTE A SOLICITUD DE: Alberto Perez

QUIEN SE IDENTIFICA CON CÓDIGO ÚNICO DE IDENTIFICACIÓN -CUI- NÚMERO: 1706337640114

Fuente: elaboración propia, realizado con aplicación web local.

Con este informe se cumple con la automatización de reportes e informes generados en la compañía de bomberos, de tal forma que ante cualquier solicitud ahora se podrá generar esta información de manera ordenada, simple y estandarizada.

CONCLUSIONES

1. Las instituciones públicas de carácter voluntario necesitan de una automatización urgente en sus procesos, tal es el caso de las entidades de socorro, las cuales deben cambiar los tradicionales métodos en papel por procesos digitales y así proveer atención expedita los requerimientos de la población en general.

2. Los sistemas web hoy en día siguen el paradigma orientado a los servicios, es necesario poder modularizar cada uno de los componentes de un sistema para así lograr una óptima administración, además de cumplir con los estándares básicos de seguridad en la transmisión y acceso a datos con información sensible.

3. Un API es uno de los mecanismos más comunes en la integración de sistemas, por medio de estas implementaciones se podrá comunicar las capas de persistencia de datos y las de presentación de datos, separando las reglas de negocio y aislando la capa de seguridad que se debe aplicar en cualquier interacción con el usuario.

4. Automatizar procesos reduce el uso de papel, lo cual es de vital importancia hoy en día, además el digitalizar la información reduce el riesgo de perdidas por factores ambientales, provee un menor impacto ecológico y el respaldo / resguardo de la información no requiere de grandes espacios físicos para tal cometido.

RECOMENDACIONES

1. Brindar apoyo técnico, velar por el correcto funcionamiento del sistema y el uso adecuado de los recursos, por tal razón se recomienda designar una comisión dentro de la compañía para dichos efectos.

2. Capacitar al personal activo ya que la resistencia al cambio siempre estará presente en la implantación de un nuevo sistema, por tal razón se recomienda que todo el personal activo de la compañía se involucre en las sesiones de capacitación del sistema SASE29.

3. Evaluar la posibilidad de contar con una infraestructura que permita el despliegue local dentro de la compañía del sistema, esto con el objetivo de a largo plazo no exponer públicamente el sistema SASE29.

4. Monitorear constantemente el crecimiento de la base de datos con el proveedor contratado, esto con el objetico de identificar cuando se esté alcanzando su capacidad máxima y no incurrir en cargos adicionales anuales por no cumplir con la cuota contratada.

REFERENCIAS

1. Ankush, (2022). Los 12 mejores *softwares* de base de datos de código abierto para su próximo proyecto. *GEEKFLARE.* Recuperado de https://geekflare.com/es/open-source-database/.

2. Bautista García, I. (30 de marzo 2021). Backend y Frontend, ¿Qué es y cómo funcionan en la programación? [Mensaje en un blog]. Recuperado de https://www.servnet.mx/blog/backend-y-frontend-partes-fundamentales-de-la-programaci%C3%B3n-de-una-aplicaci%C3%B3n-web.

3. Cobo, A. (2005). *PHP y MySql Tecnologías para el desarrollo de aplicaciones web.* Madrid, España: Ediciones Días de Santos, ISBN 84-7978-706-6. Recuperado de https://www.editdiazdesantos.com/wwwdat/pdf/9788479787066.pdf.

4. Gibb, R. (Junio 2016). What is a Web Application? [Mensaje en un blog]. Recuperado de https://www.stackpath.com/edge-academy/what-is-a-web-application/.

5. Introduction to the Angular Docs. (2021). *Angular IO*. Recuperado de https://angular.io/docs.

6. Manual de PHP. (2021), *php net*. Recuperado de https://www.php.net/manual/es/.

7. Mateu, C. (2004). *Desarrollo de aplicaciones web.* Barcelona, España: Eureca Media, SL, ISBN 84-9788-118-4. Recuperado de https://libros.metabiblioteca.org/bitstream/001/591/1/004%20Desarrollo%20de%20aplicaciones%20web.pdf.

8. MySql Documentation. (2021), *mysql.* Recuperado de https://dev.mysql.com/doc/.

9. ¿Qué es una API de REST?. (2020), *redhat*. Recuperado de https://www.redhat.com/es/topics/api/what-is-a-rest-api.

10. Representational state transfer. (2021), Wikipedia. Recuperado de https://en.wikipedia.org/wiki/Representational_state_transfer.

APÉNDICES

Apéndice 1. **Presentación y entrega de sistema SASE29**

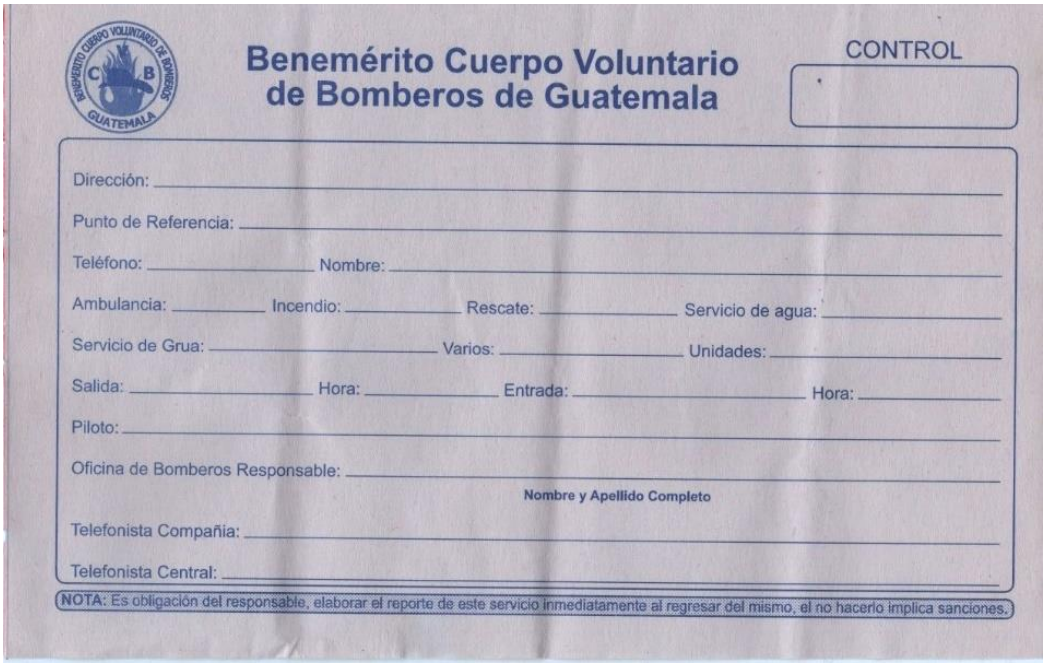
Fuente: [Fotografía de Rolando Marroquín]. (Sala de reuniones de la XXIX compañía, Amatitlán, Guatemala. 2022). Colección particular. Guatemala.

Apéndice 2. **Junta de oficiales en la primera capacitación**

Fuente: [Fotografía de Rolando Marroquín]. (Junta de oficiales en sala de reuniones de la XXIX compañía, Amatitlán, Guatemala. 2022). De izquierda a derecha: Oficial tesorero Juan Luis Pineda, Oficial secretario Manuel Esteban Chimil Ramírez, Rolando Giovanni Marroquín González, Oficial jefe de compañía Sergio Santos Guzmán, Oficial director Marco Tulio Zenteno. Colección particular. Guatemala.

ANEXOS

Anexo 1. **Boleta de solicitud de servicio**



Benemérito Cuerpo Voluntario de Bomberos de Guatemala

CONTROL

Dirección: _____

Punto de Referencia: _____

Teléfono: _____ Nombre: _____

Ambulancia: _____ Incendio: _____ Rescate: _____ Servicio de agua: _____

Servicio de Grua: _____ Varios: _____ Unidades: _____

Salida: _____ Hora: _____ Entrada: _____ Hora: _____

Piloto: _____

Oficina de Bomberos Responsable: _____

Nombre y Apellido Completo

Telefonista Compañía: _____

Telefonista Central: _____

NOTA: Es obligación del responsable, elaborar el reporte de este servicio inmediatamente al regresar del mismo, el no hacerlo implica sanciones.

Fuente: Benemérito Cuerpo Voluntario de Bomberos de Guatemala (2021). *Talonario de solicitudes de la XXIX compañía de Bomberos Voluntarios.*