# **MANUAL TÉCNICO DE DESPLIEGUE**

## **INTRODUCCIÓN**

El instructivo va dirigido a lector o usuario del aplicativo de Seguimiento. Este documento le permite conocer las funcionalidades y aspectos a tener en cuenta sobre el manejo del sistema de información de Seguimiento y los módulos de Reuniones y Seguimiento a compromisos, de acuerdo con el procedimiento de Seguimiento Estratégico a definido para la Secretaría de Gobierno, en el que se indica la forma correcta de Registrar una reunión, asignar un compromiso y completar un compromiso asignado.

La aplicación de seguimiento permite administrar y gestionar de manera segura y eficiente los compromisos derivados de reuniones y demás encuentros que conlleven una responsabilidad de un usuario responsable, a través de un ciclo o proceso el cual facilite registro de reuniones, asignación de compromisos, gestión de un compromiso, garantizando el cumplimiento y avance de acuerdo con los dispuesto por la Secretaría Gobierno.

En este documento se relaciona los requerimientos técnicos para la configuración e instalación de los elementos que constituyen el sistema de información de seguimiento.

Es importante tener en cuenta que para realizar la instalación y puesta en marcha del sistema la persona debe tener conocimientos previos en informática ya que se mencionan especificaciones mínimas de hardware y software para la correcta instalación y funcionamiento de la aplicación.

## **OBJETIVOS**

Dar a conocer la información necesaria para llevar a cabo la instalación y configuración del sistema de información:

## **OBJETIVOS ESPECÍFICOS**

* Especificar los requerimientos de la herramienta parte software
* Definir el procedimiento de instalación del aplicativo
* Detallar especificaciones de los requerimientos de software y hardware necesarios para la instalación y funcionamiento de la aplicación
* Dar a conocer el modelo entidad – relación y diccionario de datos del sistema de
* información.

## **REQUERIMIENTOS TÉCNICOS**

### **1.1 Requerimientos mínimos del sistema de información o aplicación web.**

* **Apache**: Apache/2.4.41: **servidor HTTP** que permite servir contenido de las solicitudes que vienen desde los navegadores web. Además, es una multiplataforma, es decir, que lo podrás encontrar en**Windows, Linux, Unix y Mac.**[[1]](#footnote-1)
* **PHP**: versión PHP 7.3.33 y PHP 8.0.8: PHP es un lenguaje de programación de código abierto del lado del servidor que se utiliza principalmente para crear páginas web dinámicas.
* **MySQL**: versión 15.1 Distribución 10.5.13-MariaDB o MySQL versión 8.0.25:  sistema de gestión de bases de datos relacionales de código abierto.
* **Laravel**: Laravel Installer 4.2.7:  framework PHP y de código abierto que brinda un conjunto de herramientas y recursos para crear aplicaciones modernas
* **Composer**: versión 2.5.5 2023-03-21 11:50:05 o versión 1.10.21:  manejador de paquetes para PHP que proporciona un estándar para administrar, descargar e instalar dependencias y librerías.
* **PIP**: PIP 20.0.2 - Sistema de gestión de paquetes utilizado para instalar y administrar paquetes de software escritos en Python

### **1.2 Requerimientos mínimos de hardware.**

* Procesador Intel Core o Amd
* Memoria RAM 4 GB
* Disco duro 500GB
* 20 GB de espacio en disco en el servidor web

## **Instalación y despliegue**

A continuación, se relaciona los servicios y tecnologías requeridas para completar

* 1. Instalación de PHP

sudo apt install php libapache2-mod-php php-mysql

* 1. Instalación MySQL

sudo apt install mysql-server

* 1. Instalación de Laravel y Configuración de Laravel

La instalación de Laravel la deberemos de hacer en la ruta que hemos configurado en nuestro VirtualHost, accederemos vía SSH hasta esa ruta

# cd /var/www/html

Y clonaremos el GIT de Laravel:

# git clone <https://github.com/laravel/laravel.git>

Luego moveremos los ficheros de Laravel en nuestro directorio de trabajo:

# mv laravel/\* . && mv laravel/.\* . 2> /dev/null && rmdir laravel

Finalmente, instalaremos Laravel usando Composer:

# composer install

* 1. Instalación Composer

Para instalar Composer se debe ejecutar lo siguiente:

# curl -sS https://getcomposer.org/installer | php

Una vez Composer está instalado, debemos de mover el ejecutable de Composer dentro de la ruta de nuestra máquina:

# mv composer.phar /usr/local/bin/composer

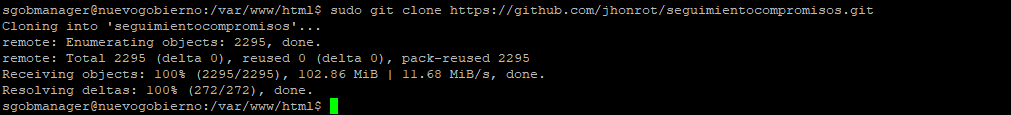
Le añadimos los permisos de ejecución:

# chmod +x /usr/local/bin/composer

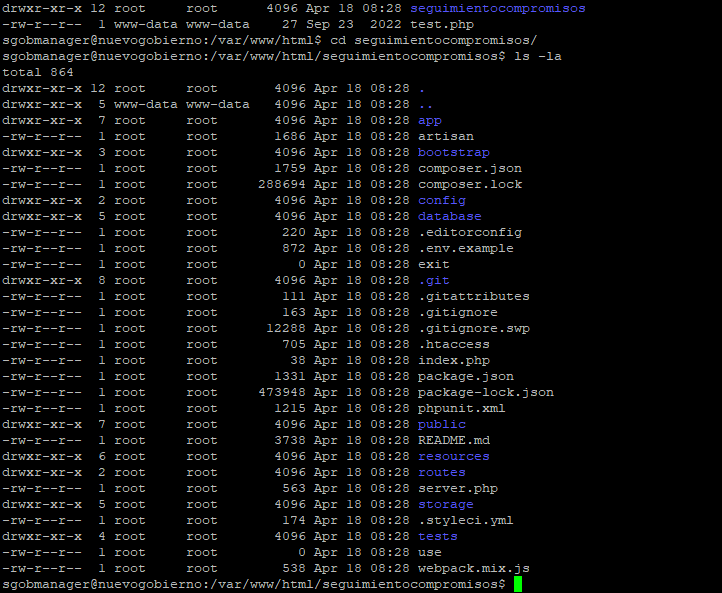
* 1. Despliegue de la aplicación

1. La de seguimiento compromisos se encuentra subido en git ***(***[***https://github.com/jhonrot/seguimientocompromisos***](https://github.com/jhonrot/seguimientocompromisos)***),*** de tal forma que podemos clonarlo en el servidor:

* sudo git clone <https://github.com/jhonrot/seguimientocompromisos.git>

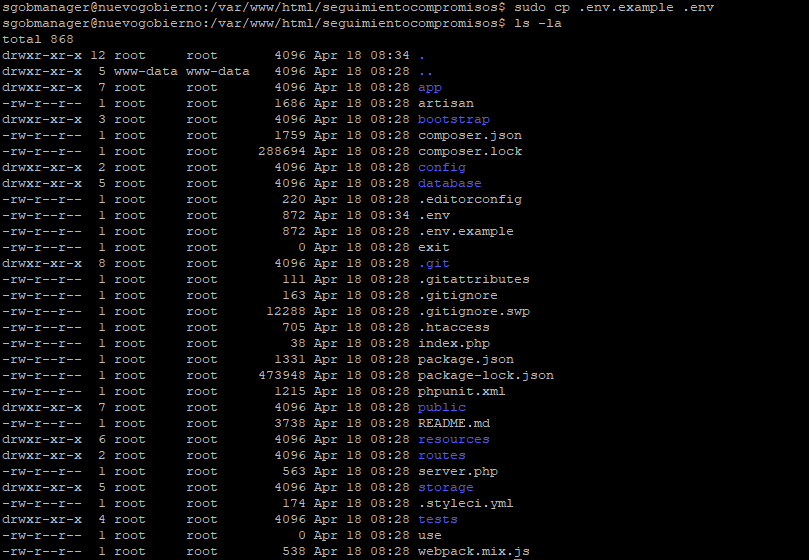


Verificamos que haya descargado e ingresamos a la carpeta



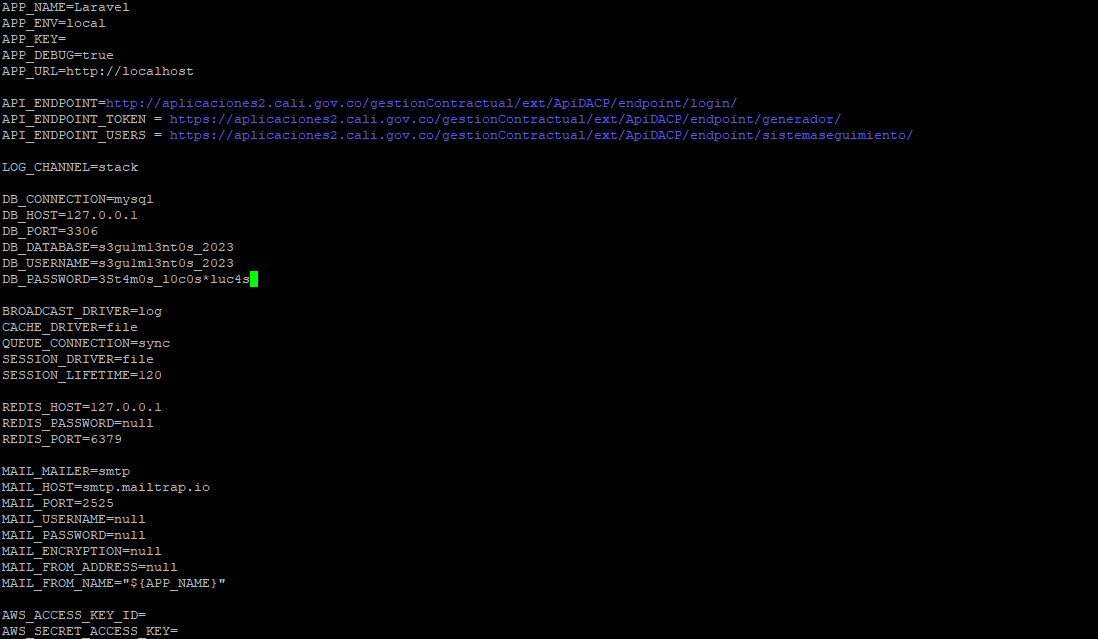
1. Paso seguido, se debe crear el archivo de variables del entorno, el cual permite configurar la base de datos, los datos del correo de notificaciones. Para crear el archivo, se debe copiar y crear el archivo **.env**, para lo cual se debe ejecutar el siguiente comando.

* sudo cp .env.example .env



1. Paso seguido, se debe editar el archivo **.env**

* sudo nano **.env**



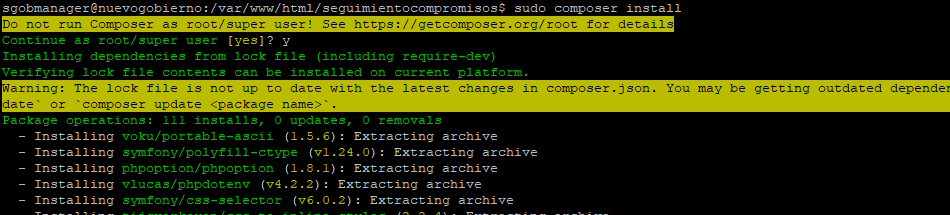
1. Paso seguido, se debe descargar e instalar la librería **node\_modules**

* sudo npm install



1. Por último, se debe descarga e instalar la librería **vendor,**

* sudo composer install



1. **Descripción general del sistema**

Se encuentra basado en la arquitectura de desarrollo MVC (modelo, vista y controlador), en el que cada componente se encarga de un determinado proceso en el manejo de la información.

La aplicación en su parte software se divide en dos:

* **Back End:** Perteneciente a procesar los datos de los registros de los compromisos derivados de encuentros y/o reuniones
* **FrontEnd:** Perteneciente a recibir datos enviados por el Backend por medio de Sockets y de visualizarlos de una manera entendible y coherente al usuario final de la aplicación.

1. **Modelo entidad relación**

El modelo entidad relacional o base de datos cuenta con la siguientes entidades y atributos.

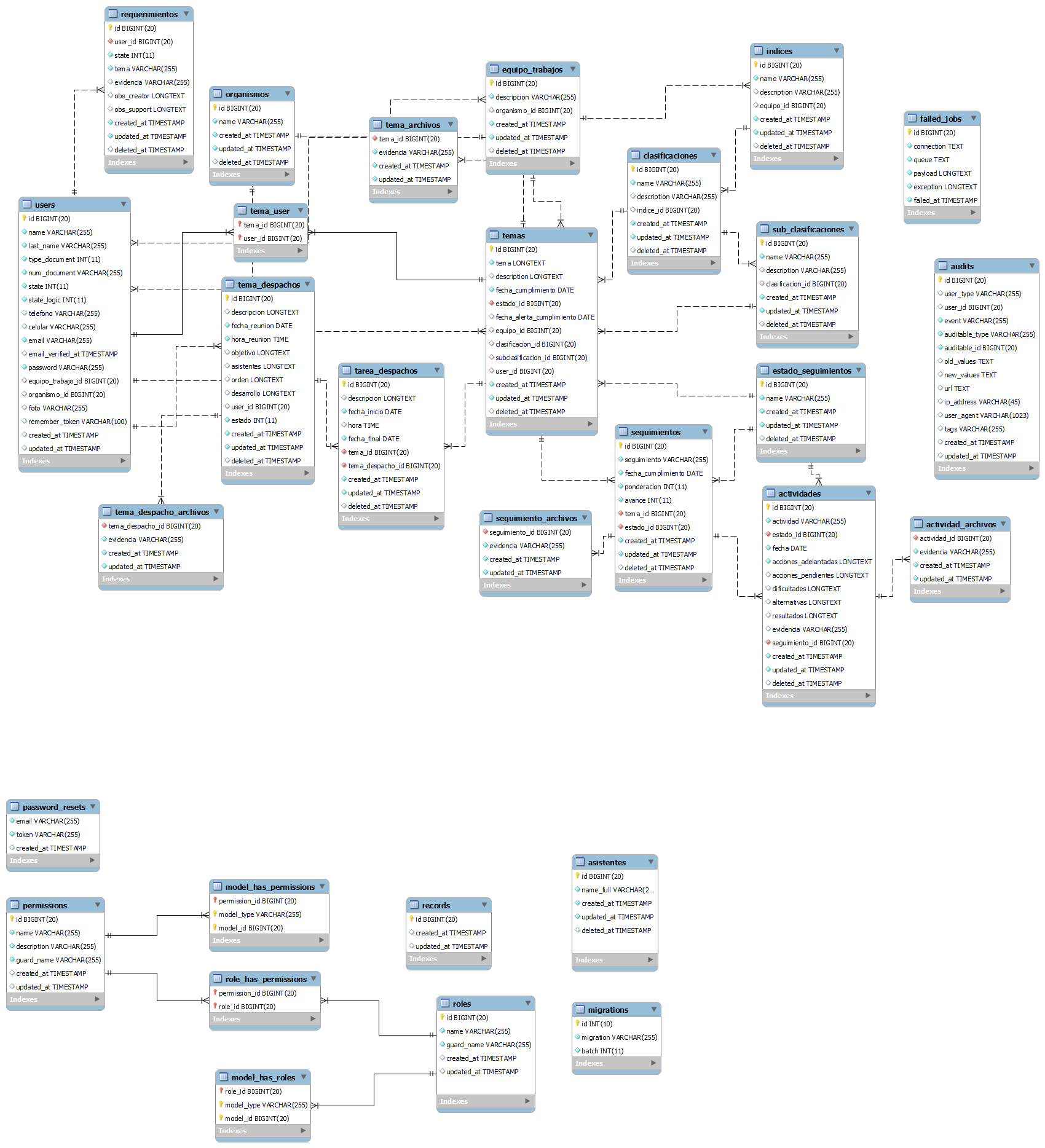
****

Imagen 1 Modelo Entidad Relación

**4.1 Diccionario de datos**

A continuación, se relaciona el diccionario de datos, descripción de las entidades y atributos que conforman el modelo de datos.

|  |  |
| --- | --- |
| Entidades | Descripción |
| CREATE TABLE IF NOT EXISTS `equipo\_trabajo` (    `idEquipo` int(11) NOT NULL AUTO\_INCREMENT,    `descEquipoTrabajo` varchar(60) NOT NULL,    PRIMARY KEY (`idEquipo`)  ) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO\_INCREMENT=1 ; | Tabla que permitirá almacenar los equipos de trabajo de la Secretaría de Gobierno. |
| CREATE TABLE IF NOT EXISTS `estadoseguimiento` (    `idEstadoSeguimiento` int(11) NOT NULL AUTO\_INCREMENT,    `EstadoSeguimiento` varchar(45) NOT NULL,    PRIMARY KEY (`idEstadoSeguimiento`)  ) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO\_INCREMENT=1 ; | La tabla estado seguimiento corresponde a la entidad que permitirá relacionar un estado de seguimiento a un tema. La tabla almacenará los estados definidos por la Secretaría de Gobierno. |
| CREATE TABLE IF NOT EXISTS `organismo` (    `idOrganismo` int(11) NOT NULL AUTO\_INCREMENT,    `nombreOrganismo` varchar(100) NOT NULL,    PRIMARY KEY (`idOrganismo`)  ) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO\_INCREMENT=1 ; | La table organismo permitirá almacenar los nombres de los organismo y dependencias de la alcaldía de Santiago de cali. La entidad se relacionara con los tablas Responsable y Tema. |
| CREATE TABLE IF NOT EXISTS `perfil\_usuario` (    `idPerfil\_Usuario` int(11) NOT NULL AUTO\_INCREMENT,    `perfilUsuario` varchar(45) NOT NULL,    PRIMARY KEY (`idPerfil\_Usuario`)  ) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO\_INCREMENT=1 ; | Tabla que permitirá almacenar los perfiles de usuarios definidos por la secretaría de gobierno, para llevar a cabo un seguimiento. |
| CREATE TABLE IF NOT EXISTS `prioridad` (    `idPrioridad` int(11) NOT NULL,    `Prioridad` varchar(45) NOT NULL,    PRIMARY KEY (`idPrioridad`)  ) ENGINE=InnoDB DEFAULT CHARSET=latin1; | Tabla que almacenará la clasificación de prioridad para un tema. |
| CREATE TABLE IF NOT EXISTS `responsable` (    `idResponsable` int(11) NOT NULL AUTO\_INCREMENT,    `nombre` varchar(45) NOT NULL,    `apellidos` varchar(45) NOT NULL,    `email` varchar(45) NOT NULL,    `Perfil\_Usuario\_idPerfil\_Usuario` int(11) NOT NULL,    `Equipo\_Trabajo\_idEquipo` int(11) NOT NULL,    `Organismo\_idOrganismo` int(11) NOT NULL,    PRIMARY KEY (`idResponsable`,`Perfil\_Usuario\_idPerfil\_Usuario`,`Equipo\_Trabajo\_idEquipo`,`Organismo\_idOrganismo`),    KEY `fk\_Usuario\_Perfil\_Usuario1` (`Perfil\_Usuario\_idPerfil\_Usuario`),    KEY `fk\_Usuario\_Equipo\_Trabajo1` (`Equipo\_Trabajo\_idEquipo`),    KEY `fk\_Persona\_Organismo1` (`Organismo\_idOrganismo`)  ) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO\_INCREMENT=1 ; | Tabla que relaciona los responsables de un tema. La tabla permitirá los responsables de cada uno de los temas. |
| CREATE TABLE IF NOT EXISTS `responsable\_has\_tema` (    `Responsable\_idResponsable` int(11) NOT NULL,    `Responsable\_Perfil\_Usuario\_idPerfil\_Usuario` int(11) NOT NULL,    `Responsable\_Equipo\_Trabajo\_idEquipo` int(11) NOT NULL,    `Responsable\_Organismo\_idOrganismo` int(11) NOT NULL,    `Tema\_idTema` int(11) NOT NULL,    `Tema\_Prioridad\_idPrioridad` int(11) NOT NULL,    PRIMARY KEY (`Responsable\_idResponsable`,`Responsable\_Perfil\_Usuario\_idPerfil\_Usuario`,`Responsable\_Equipo\_Trabajo\_idEquipo`,`Responsable\_Organismo\_idOrganismo`,`Tema\_idTema`,`Tema\_Prioridad\_idPrioridad`),    KEY `fk\_Responsable\_has\_Tema\_Tema1` (`Tema\_idTema`,`Tema\_Prioridad\_idPrioridad`)  ) ENGINE=InnoDB DEFAULT CHARSET=latin1; | La tabla corresponde a la entidad que permitirá almacenar y relacionar todos los temas registrados por un responsable. Cada tema puede tener entre un responsable y varios responsables, y un tema podrá tener por organismo un responsable, |
| CREATE TABLE IF NOT EXISTS `seguimiento` (    `idEstadoSeguimiento` int(11) NOT NULL AUTO\_INCREMENT,    `fechaSeguimiento` date NOT NULL,    `EstadoSeguimiento\_idEstadoSeguimiento` int(11) NOT NULL,    `accionesAdelantadas` varchar(200) NOT NULL,    `accionesPendienttes` varchar(200) NOT NULL,    `dificultadesPresentadas` varchar(200) DEFAULT NULL,    `alternativasSolucion` varchar(200) DEFAULT NULL,    `porcentajeAvance` int(11) NOT NULL,    `porcentajeRestante` int(11) NOT NULL,    `evidencia` blob,    `Responsable\_has\_Tema\_Responsable\_idResponsable` int(11) NOT NULL,    `Responsable\_has\_Tema\_Responsable\_Perfil\_Usuario\_idPerfil\_Usuario` int(11) NOT NULL,    `Responsable\_has\_Tema\_Responsable\_Equipo\_Trabajo\_idEquipo` int(11) NOT NULL,    `Responsable\_has\_Tema\_Responsable\_Organismo\_idOrganismo` int(11) NOT NULL,    `Responsable\_has\_Tema\_Tema\_idTema` int(11) NOT NULL,    `Responsable\_has\_Tema\_Tema\_Prioridad\_idPrioridad` int(11) NOT NULL,    PRIMARY KEY (`idEstadoSeguimiento`,`EstadoSeguimiento\_idEstadoSeguimiento`,`Responsable\_has\_Tema\_Responsable\_idResponsable`,`Responsable\_has\_Tema\_Responsable\_Perfil\_Usuario\_idPerfil\_Usuario`,`Responsable\_has\_Tema\_Responsable\_Equipo\_Trabajo\_idEquipo`,`Responsable\_has\_Tema\_Responsable\_Organismo\_idOrganismo`,`Responsable\_has\_Tema\_Tema\_idTema`,`Responsable\_has\_Tema\_Tema\_Prioridad\_idPrioridad`),    KEY `fk\_Seguimiento\_EstadoSeguimiento1` (`EstadoSeguimiento\_idEstadoSeguimiento`),    KEY `fk\_Seguimiento\_Responsable\_has\_Tema1` (`Responsable\_has\_Tema\_Responsable\_idResponsable`,`Responsable\_has\_Tema\_Responsable\_Perfil\_Usuario\_idPerfil\_Usuario`,`Responsable\_has\_Tema\_Responsable\_Equipo\_Trabajo\_idEquipo`,`Responsable\_has\_Tema\_Responsable\_Organismo\_idOrganismo`,`Responsable\_has\_Tema\_Tema\_idTema`,`Responsable\_has\_Tema\_Tema\_Prioridad\_idPrioridad`)  ) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO\_INCREMENT=1 ; | Tabla seguimiento permitirá almacenar los registros derivados a un seguimiento de un tema ya registrado. Un tema puede tener muchos seguimientos. |
| CREATE TABLE IF NOT EXISTS `tema` (    `idTema` int(11) NOT NULL AUTO\_INCREMENT,    `descTema` varchar(200) NOT NULL,    `prioridad` varchar(40) NOT NULL,    `clasificacion` varchar(100) NOT NULL,    `observacion` varchar(200) DEFAULT NULL,    `fechaInicio` date NOT NULL,    `fechaMaxCumplimiento` date DEFAULT NULL,    `evidencia` blob,    `plazoDias` int(11) NOT NULL,    `diasRestantes` int(11) NOT NULL,    `alerta` varchar(45) DEFAULT NULL,    `Prioridad\_idPrioridad` int(11) NOT NULL,    PRIMARY KEY (`idTema`,`Prioridad\_idPrioridad`),    KEY `fk\_Tema\_Prioridad1` (`Prioridad\_idPrioridad`)  ) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO\_INCREMENT=1 ; | La tabla tema corresponde a la entidad que permitirá registrar todos los temas bajo una clasificación, un responsable y un organismo. |

1. https://www.crehana.com/blog/transformacion-digital/servidor-apache/ [↑](#footnote-ref-1)