Manual Técnico - Sistema de Gestión de Tickets (SGTS)

Versión del Software: 1.0 **Tecnología:** CodeIgniter 4 / PHP 8.2 / MySQL **Fecha:** 10 de Diciembre de 2025

1. Descripción General

El SGTS es una aplicación web basada en el patrón de arquitectura **Modelo-Vista-Controlador (MVC)** diseñada para la administración centralizada de incidentes. Utiliza **CodeIgniter 4** como framework backend y **Bootstrap 5 / CoreUI** para el frontend.

2. Requisitos del Sistema (Entorno)

Para desplegar la aplicación, el servidor debe cumplir con:

- **PHP:** Versión 8.1 o superior (Recomendado 8.2).
- **Extensiones PHP obligatorias:**
  - `intl` (Para CodeIgniter).
  - `gd` (Para PhpSpreadsheet/Gráficos).
  - `mbstring`, `json`, `mysql`, `xml`.
- **Base de Datos:** MySQL 5.7+ o MariaDB 10.3+.
- **Servidor Web:** Apache (con mod_rewrite habilitado) o Nginx.
- **Gestor de Dependencias:** Composer 2.x.

3. Instalación y Despliegue

3.1 Configuración Inicial

1. Clonar el repositorio o descomprimir el código fuente en la carpeta pública del servidor (`htdocs` o https://github.com/FGFERNAN/SGTS).
2. Abrir una terminal en la raíz del proyecto y ejecutar:
   `composer install`
3. *Esto instalará CodeIgniter, Dompdf y PhpSpreadsheet.*

3.2 Configuración de Entorno (.env)

1. Renombrar el archivo `env` a `.env`.
2. Configurar la URL base y la base de datos:
   CI_ENVIRONMENT = development
   app.baseURL = 'http://localhost/sgts/public/'

   database.default.hostname = localhost
   database.default.database = sgts
   database.default.username = root
   database.default.password =
   database.default.DBDriver = MySQLi

3.3 Base de Datos

1. Crear una base de datos vacía llamada `sgts`.
2. Importar el archivo SQL suministrado (`SGTS/docs/Diseño Conceptual/sgts.sql`).
   - Esto creará las tablas: `usuarios`, `tickets`, `historial`, `notificaciones`, etc.

4. Arquitectura del Proyecto

4.1 Estructura de Directorios Clave

- `app/Controllers/`: Lógica de negocio dividida por roles (`Admin`, `Tecnico`, `Cliente`).
- `app/Models/`: Interacción con la base de datos. Se utiliza **Query Builder** y Entidades.
- `app/Views/`: Interfaces de usuario renderizadas en HTML.
    - `layouts/`: Plantillas maestras (`main_template.php`).
- `app/Helpers/`: Helpers personalizados (ej. `notificaciones_helper.php`).
- `public/assets/`: Recursos estáticos (CSS, JS, Imágenes, Librerías de terceros como Chart.js).

4.2 Librerías de Terceros

El proyecto integra las siguientes librerías vía Composer:

- **Dompdf:** Generación de reportes en PDF.
- **PhpSpreadsheet:** Exportación de datos a Excel (.xlsx).
- **Chart.js:** (Frontend) Renderizado de gráficos estadísticos en el Dashboard.

5. Módulos Principales y Lógica

5.1 Seguridad y Autenticación

- Se utiliza `password_hash()` (Bcrypt) para el almacenamiento de contraseñas.
- Filtros de Ruta (`app/Filters/`):
    - `AuthFilter`: Verifica sesión activa.
    - `RoleFilter`: Restringe acceso según `id_rol` (1=Admin, 2=Técnico, 3=Cliente).
- Protección contra **CSRF** activada en todos los formularios.

5.2 Sistema de Notificaciones

- **Backend:** Modelo `NotificacionModel` y tabla `notificaciones`.
- **Frontend:** Helper `obtener_notificaciones_usuario()` inyectado en el Navbar.
- **Flujo:** Al asignar un ticket, el controlador crea el registro. Al hacer clic en la campana, se marca como `leido=1` vía AJAX/Redirección.

5.3 Reportes Dinámicos

- Los reportes utilizan parámetros `GET` en la URL para permitir compartir enlaces filtrados.
- La exportación a PDF/Excel reutiliza la lógica de filtros del controlador para garantizar la consistencia de los datos.

6. Diagrama Entidad-Relación (MER)