Telefónica

# Manual Técnico

## Sistema de Gestión de Contratos de Arrendamiento

Tipo interno

31 de enero de 2025

## Control de cambios

| Versión | Fecha de actualización | Autor del cambio | Descripción de la actualización |
|---|---|---|---|
| 1.0 | 3/10/2024 | Hamilton Leon | Inicio del documento |
| | | | |

## Aprobaciones del documento

| Fecha | Nombre | Cargo | Firma |
|---|---|---|---|
| | | | |
| | | | |

# 1. Tabla de contenido

# 2. Introducción

## 2.1 Breve descripción del sistema.

El Sistema de Gestión de Contratos es una solución diseñada para la Gerencia de Planificación e Inmuebles de Telefónica Venezolana C.A. Facilita el registro, organización y clasificación eficiente de los contratos de arrendamiento de los locales de la empresa, optimizando la administración de esta información crítica.

Además del registro, el sistema ofrece herramientas para generar reportes y alertas sobre contratos próximos a vencer, permitiendo una gestión proactiva y la toma de decisiones estratégicas. Su interfaz intuitiva mejora la accesibilidad a la información y asegura que los datos estén siempre actualizados.

## 2.2 Objetivo del documento.

Este manual proporciona una guía clara y detallada para el despliegue del sistema de gestión de contratos en servidores Linux. Su objetivo es asegurar una instalación eficiente y estandarizada, minimizando errores y garantizando el correcto funcionamiento del sistema.

El documento abarca la configuración del entorno, instalación de dependencias, configuración del servidor web, migración de la base de datos y verificación de funcionalidades clave.

## 2.3 Audiencia.

Este manual está dirigido a los siguientes grupos y roles:

**Administradores de Sistemas**
Responsables de la instalación, configuración y mantenimiento de los servidores donde se desplegará el sistema. Incluye la gestión de dependencias, configuración del servidor web y monitoreo de recursos del sistema.

**Desarrolladores y Equipos Técnicos**
Encargados de garantizar la correcta integración del código fuente en los entornos de desarrollo, pruebas y producción, así como de solucionar problemas técnicos durante el despliegue.

**Equipos de Soporte Técnico**
Enfocados en el mantenimiento del sistema una vez desplegado, incluyendo la gestión de actualizaciones, resolución de errores y respaldo de datos.

**Auditores Técnicos**
Encargados de revisar y validar que el despliegue se realice conforme a las mejores prácticas y los estándares establecidos por la organización.

Este documento asume que los lectores poseen conocimientos básicos sobre administración de servidores, sistemas operativos Linux y/o Windows Server, bases de datos MySQL, y herramientas relacionadas como PHP y Laravel.

# 3. Requisitos Previos

## 3.1 Hardware

- **Servidor de Producción:**

  **Almacenamiento:** 500 gb (SSD recomendado para mayor velocidad).

- **Servidor de Desarrollo:**

  Para el desarrollo y pruebas del sistema se utilizó un equipo con las siguientes especificaciones:

  **CPU**: Intel i7-8665U.

  **RAM**: 16 GB.

  **Almacenamiento**: SSD de 500 GB.

## 3.2 Software

El sistema requiere los siguientes componentes y dependencias:

- **Para Producción:**

**Sistema Operativo:**

Linux (Debian 11 o superior, CentOS 8 o superior).

**Servidor Web:**

Apache 2.4 o superior.

Nginx 1.18 o superior (opcional).

**Lenguaje de Programación:** PHP 8.2 o superior.

**Base de Datos:** MySQL 8.0 o superior.

**Composer:** Para la gestión de dependencias de PHP.

- **Para Desarrollo:**

Para el desarrollo y pruebas del sistema se utilizaron las siguientes tecnologías de software

**Sistema Operativo:** Windows 10.

**Entorno de Desarrollo:** XAMPP (Apache, PHP 8.2, MySQL).

**Editor de Código:** Visual Studio Code

## 3.3 Acceso

Para la instalación y despliegue, es necesario garantizar los siguientes accesos:

**Acceso a Internet:**

Apertura del puerto **443** hacia la IP **167.114.128.168** (repositorio oficial de PHP) para la instalación de librerías y actualizaciones de dependencias mediante Composer.

**Credenciales de Bases de Datos:** Usuario y contraseña para MySQL.

**Acceso SSH:** Si se utiliza un servidor Linux, habilitar SSH para la administración remota.

## 3.4 Repositorios y Archivos

**Dependencias:** Archivo composer.json para instalar librerías necesarias.

**Configuraciones:** Archivo .env con las variables de entorno específicas para el entorno de producción.

# 4. Despliegue en Linux

### 1. Instalación del Entorno

### 1.1 Configuración inicial del servidor

Actualizar el sistema operativo:

```
sudo apt update && sudo apt upgrade -y
```

Configurar el hostname del servidor y ajustar la zona horaria:

```
sudo hostnamectl set-hostname nombre-del-servidor
sudo timedatectl set-timezone America/Caracas
```

### 1.2 Instalación de dependencias

**Instalar Apache o Nginx:**

Apache:

```
sudo apt install apache2 -y
```

Nginx:

```
sudo apt install nginx -y
```

**Instalar PHP y extensiones requeridas:**

```
sudo apt install php php-mysql php-xml php-mbstring php-zip curl unzip php-gd -y
```

**Instalar MySQL:**

```
sudo apt install mysql-server -y
sudo mysql_secure_installation
```

**Instalar Composer:**

```
php -r "copy('https://getcomposer.org/installer', 'composer-setup.php');"

php -r "if (hash_file('sha384', 'composer-setup.php') ===
'dac665fdc30fdd8ec78b38b9800061b4150413ff2e3b6f88543c636f7cd84f6db9189d43a81e5503cda
447da73c7e5b6') { echo 'Installer verified'; } else { echo 'Installer corrupt';
unlink('composer-setup.php'); } echo PHP_EOL;"

php composer-setup.php

php -r "unlink('composer-setup.php');"

sudo mv composer.phar /usr/local/bin/composer
```

**Instalar node.js:**

```
curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.40.1/install.sh | bash
```

Cierra la terminal y vuelve a abrir.

```
nvm install 22
```

**1.3 Configuración del servidor web**

**Configurar Apache o Nginx para servir el proyecto:**

**Apache**: Crear un archivo de configuración en /etc/apache2/sites-available/nombre-del-sitio.conf

**Nginx**: Crear un archivo de configuración en /etc/nginx/sites-available/nombre-del-sitio


nombre_sistema.conf

**Activar la configuración y reiniciar el servicio web:**

**Apache:**

```
sudo a2ensite nombre-del-sitio.conf

sudo systemctl reload apache2
```

**Nginx:**

```
sudo ln -s /etc/nginx/sites-available/nombre-del-sitio /etc/nginx/sites-enabled/

sudo systemctl reload nginx
```

**2. Configuración del Sistema**

**2.1 Clonar o transferir la carpeta del sistema en el servidor**

**2.2 Configuración de variables de entorno (.env)**

```
sudo nano .env
```


.env

Configurar los valores como base de datos, credenciales y variables específicas del entorno.

### 2.3 Configuración de permisos de carpetas

Establecer los permisos correctos para las carpetas de almacenamiento y caché:

```
sudo chown -R www-data:www-data /var/www/nombre-del-sitio

sudo chmod -R 775 /var/www/nombre-del-sitio/storage /var/www/nombre-del-sitio/bootstrap/cache
```

## 3. Migración y Seeds de la Base de Datos

### 3.1 Ejecución de migraciones

Ejecutar las migraciones para crear las tablas necesarias:

```
php artisan migrate
```

### 3.2 Ejecución de seeders

Poblar la base de datos con datos iniciales usando los seeders:

```
php artisan db:seed
```

## 4. Despliegue del Código

### 4.1 Instalación de dependencias

Instalar las dependencias PHP con Composer:

```
composer install --no-dev --optimize-autoloader
```

Instalar las dependencias del front-end:

```
npm install && npm run build
```

## 5. Pruebas Post-Despliegue

### 5.1 Comprobación del estado del servidor web

Asegurarse de que el servidor web esté en ejecución:

**Apache:**

```
sudo systemctl status apache2
```

**Nginx:**

```
sudo systemctl status nginx
```

### 5.2 Verificación de los logs del sistema

Revisar los logs para identificar errores:

**Apache**: /var/log/apache2/error.log

**Nginx:** /var/log/nginx/error.log

**Laravel:** storage/logs/laravel.log

### 5.3 Validación de URLs clave

Probar las rutas principales del sistema en un navegador para verificar su correcto funcionamiento.

Confirmar que las funcionalidades críticas (inicio de sesión, registro, generación de reportes) operan como se espera.

# 5. Gestión de Errores Comunes

## 5.1 Errores frecuentes durante la instalación.

### 5.1.1 Error de instalación MySQL

Si la instalación de MySQL ocasiona algún error, debido a diferentes factores. Procederemos con la instalación de maríaDB como SMBD

```
sudo apt install mariadb.server -y
```

Se hace la instalación segura

```
sudo mysql_secure_installation
```

### 5.2 No se almacenan los archivos relacionados con los contratos

**Causa:** Este error ocurre porque no se realiza correctamente el acceso directo de la carpeta en donde se almacenan los archivos (storage/contratos) a la carpeta public

**Solución:** Realizar el acceso directo de manera manual, en caso de que después de realizar el

```
php artisan storage:link
```

No funcione se debe de realizar el acceso directo de manera manual con el comando

```
ln -s /var/www/html/arrilo/storage/app/contracts
      /var/www/html/arrilo/public/storage
```

Modificando la ruta de las carpetas, con la ruta real en donde se haya almacenado el sistema en producción

# 6. Actualizaciones y Mantenimiento

## 6.1 Procedimientos para actualizar el sistema.

Las actualizaciones del sistema se realizarán siguiendo estos pasos:

**1-. Copia de seguridad:** Antes de cualquier actualización, es fundamental realizar una copia de seguridad completa del sistema, incluyendo la base de datos y los archivos del proyecto. Esto te permitirá restaurar el sistema en caso de cualquier problema durante la actualización.

**2-. Modo de mantenimiento:** Activa el modo de mantenimiento del sistema para evitar que los usuarios accedan mientras se realiza la actualización. Puedes hacerlo ejecutando el siguiente comando en la terminal:

```
php artisan down
```

**3-. Actualización de dependencias:** Actualiza las dependencias del proyecto a través de Composer. Ejecuta el siguiente comando para actualizar a las versiones más recientes compatibles:

```
composer update
```

**4-. Migraciones de la base de datos:** Si la actualización incluye cambios en la estructura de la base de datos, ejecuta las migraciones para aplicar estos cambios. Utiliza el siguiente comando:

```
php artisan migrate
```

**5-. Limpieza de caché:** Limpia la caché del sistema para asegurar que los cambios se apliquen correctamente. Ejecuta los siguientes comandos:

```
php artisan cache:clear
php artisan view:clear
```

**8-. Desactivar el modo de mantenimiento:** Una vez que hayas verificado que todo funciona correctamente, desactiva el modo de mantenimiento:

```
php artisan up
```

**7-. Pruebas:** Realiza pruebas exhaustivas del sistema para verificar que todas las funcionalidades funcionan correctamente después de la actualización.

### 6.2 Consideraciones para mantener las dependencias actualizadas.

• **Frecuencia:** Se recomienda revisar y actualizar las dependencias del sistema de forma regular, por ejemplo, una vez al mes o según la frecuencia de publicación de nuevas versiones.

• **Versiones:** Presta atención a las versiones de las dependencias que se actualizan. Asegúrate de que las nuevas versiones sean compatibles con el resto del sistema y no introduzcan conflictos.

• **Notas de la versión:** Antes de actualizar una dependencia, consulta las notas de la versión para conocer los cambios y posibles problemas de compatibilidad.

• **Pruebas:** Después de actualizar las dependencias, realiza pruebas exhaustivas del sistema para verificar que todo funcione correctamente.

## 7. Respaldo y Recuperación

### 7.1 Procedimientos para respaldar datos.

Para mayor seguridad y facilidad, hemos automatizado el proceso de respaldo tanto del sistema como de la base de datos. Esto se realiza mediante un programa (script) que se ejecuta automáticamente todos los días a las 6 PM, gracias a una función llamada "crontab".


backup_arrilo_script

Este programa realiza las siguientes acciones:

1. **Copia de seguridad de los archivos del sistema:**
    - Crea una carpeta con la fecha del día (ej: 25-08-2024) dentro de la carpeta "respaldos_arrilo".
    - Copia todos los archivos del sistema a esta carpeta.
    - Comprime la carpeta en un archivo llamado "respaldo_aplicacion.tar.gz" para ahorrar espacio.

2. **Copia de seguridad de la base de datos:**
    - Crea un archivo llamado "respaldo_BD.sql" dentro de la carpeta con la fecha del día.
    - Copia toda la información de la base de datos a este archivo.

## 7.2 Pasos para restaurar un respaldo.

En caso de ser necesario, puedes restaurar el sistema siguiendo estos pasos:

**Ubica la carpeta de respaldos:**

Abre la carpeta "respaldos_arrilo" y busca la carpeta con la fecha del respaldo que deseas utilizar.

**Descomprime los archivos del sistema:**

Descomprime el archivo "respaldo_aplicacion.tar.gz".

Copia los archivos y carpetas resultantes a la ubicación principal del sistema, reemplazando los archivos existentes.

**Restaura la base de datos:**

Abre el programa de gestión de bases de datos (SMBD).

Elimina la base de datos actual.

Importa el archivo "respaldo_BD.sql" para restaurar la base de datos a su estado anterior.

**Importante:**

Te recomendamos eliminar la base de datos antes de restaurarla para evitar problemas de compatibilidad y datos duplicados.


**www.telefonica.com**