**Manual de Mantenimiento**

El **Manual de Mantenimiento** es un documento que explica cómo realizar tareas de mantenimiento para garantizar que el sistema funcione correctamente a lo largo del tiempo. En tu caso, el mantenimiento se enfoca en:

* ✅ Base de datos MySQL (limpieza, copias de seguridad y optimización).
* ✅ Aplicación en Python (actualización de paquetes y corrección de errores).
* ✅ Servidor Local (Docker con Nginx y Flask) (mantener contenedores y servicios en buen estado).

**1. Introducción**

Este manual describe los procedimientos necesarios para el mantenimiento del sistema de visitas, incluyendo la base de datos MySQL, la aplicación Python y el servidor local (configurado con Docker).

**2. Mantenimiento Preventivo**

El mantenimiento preventivo evita problemas antes de que ocurran. Se recomienda realizarlo periódicamente.

**2.1 Copia de Seguridad de la Base de Datos**

Realiza un respaldo de la base de datos **Visitas** cada semana.

1. Abre MySQL Workbench o phpMyAdmin en http://localhost:8080/ (si usas phpMyAdmin en Docker).
2. Conéctate al contenedor MySQL y selecciona la base de datos **Visitas**.
3. Haz clic en **Data Export** (si usas MySQL Workbench) o en **Exportar** (si usas phpMyAdmin).
4. Selecciona el formato **SQL** y guarda el archivo en una carpeta segura (por ejemplo, ./backups/).

Si estás utilizando Docker, también puedes realizar copias de seguridad automáticas utilizando el contenedor **mysql-backup**, que se ejecuta todos los días:

docker exec mysql-backup /bin/sh -c "mysqldump -h mysql -u${MYSQL\_USER} -p${MYSQL\_PASSWORD} ${MYSQL\_DATABASE} > /backups/backup\_$(date +%Y%m%d\_%H%M%S).sql"

**2.2 Actualización de Paquetes de Python**

Cada mes, revisa y actualiza los paquetes de Python para asegurar que el sistema esté actualizado y funcionando correctamente:

pip install --upgrade -r requirements.txt

Este comando actualizará todos los paquetes listados en el archivo requirements.txt del proyecto.

**2.3 Verificación del Servidor Local (Docker y MYSQL Worbench)**

Si usas Docker para tu servidor, sigue estos pasos:

1. Verifica que los contenedores de MySQL, Flask y Nginx estén en ejecución con:

docker ps

1. Si el contenedor de MySQL no está en ejecución, reinícialo:

docker restart mysql-server

1. Si el contenedor de Flask no está en ejecución, reinícialo:

docker restart flask-app

**3. Mantenimiento Correctivo**

Si el sistema presenta fallos, sigue estos pasos.

**3.1 Errores en la Base de Datos**

Si la base de datos no responde, revisa si las tablas están dañadas en MySQL.

Para reparar una tabla, usa este comando en MySQL:

sql

REPAIR TABLE visitas;

**3.2 Errores en la Aplicación Python**

Si la aplicación Python no arranca, revisa el mensaje de error en la terminal.

**Error de conexión a MySQL (mysql.connector.errors.ProgrammingError: 1045):**

* Revisa que el usuario y la contraseña en el archivo config.py sean correctos.

**Error 500 en Flask:**

* Ejecuta Flask en modo depuración para obtener más detalles sobre el error:

python visitas.py --debug

**3.3 Reinicio de Servicios**

Si la aplicación o MySQL fallan, intenta lo siguiente:

**Reiniciar MYSQL (si usas MYSQL fuera de Docker):**

1. Cierra MYSQL Worbench completamente.
2. Ábrelo de nuevo y activa comando docker-compose

**Reiniciar Flask (si usas Docker con Flask):**

1. Detén el contenedor de Flask:

docker stop flask-app

1. Ejecuta nuevamente el contenedor de Flask:

docker start flask-app