Universidad San Carlos de Guatemala

Ingeniería Ciencias en Sistemas

Software Avanzado

# Proyecto : API CMDB

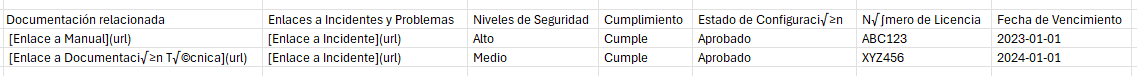
Nombre : Andy Ezequiel Sanic Tiul

Carnet : 202006699

Link a repositorio : <https://github.com/AndyST70/Practicas-SA-A-202006699/blob/main/Tareas%20vacas/tarea%203/cmdb_api/routes/ci_routes.py>

# Estructura básica del concepto





## Tecnologias

Se utilizo flask por ser ligero, fácil de configurar y suficiente para una API REST académica.

Tener instalado lo siguiente :

|  |
| --- |
| Tecnologias |

* JWT
* mysql-connector-python
* python-dotenv
* pandas
* Flask

## Base de datos - MySQL

Se utilizo un modelo relacional que incluye lo siguiente:

* ci: Información base de los items de configuración.
* ambiente: Etiquetado de los CIs por entorno (requisito funcional).
* arbol: Para representar relaciones tipo padre-hijo entre CIs.
* cambio, bitacora\_cambio, documento: Para registrar auditoría básica de cambios, cumpliendo con el requisito de trazabilidad.

Se incluyo un /reset\_tablas para pruebas rápidas durante el desarrollo.

## Diagrama lógico

Diagrama

El contenido generado por IA puede ser incorrecto.

## Entidad Relación

Interfaz de usuario gráfica, Diagrama

El contenido generado por IA puede ser incorrecto.

## Api

|  |  |  |
| --- | --- | --- |
| Método | Ruta | Descripción |
| POST | /login | Autenticación y generación de token |
| GET | /ci | Listado de todos los CIs |
| POST | /ci | Crear nuevo CI |
| GET | /ci/<id> | Obtener un CI por ID |
| DELETE | /ci/<id> | Eliminar CI |
| PUT | /ci/<int:id> | Actualizar ci |
| POST | /cambio | Crear cambio con bitácora y doc |
| PUT | /cambio/<id> | Actualizar un cambio |
| POST | /reset\_tablas | Limpia todas las tablas (dev) |
| POST | /cargar\_ci | Carga masiva desde Excel |

## Estructura de arquitectura

|  |
| --- |
| cmdb\_api/  |── config/  | └── db.py  |── routes/  | └── ci\_routes.py  |── utils/  | └── utils.py  |── main.py  |── .env  |── requirements.txt  |── CMDB.xlsx  |── README.md |

## Como ejecutar

* Clona el repo
* Crear .env
* Instalar dependencias pip install -r requeriments.txt
* Corre con pyton main.py
* Acceder con postman o navegador

## Roles por ambiente

|  |  |  |
| --- | --- | --- |
| Ambiente | Descripción | Responsabilidad Principal |
| DEV | Desarrollo | Crear, probar y modificar CIs en fase inicial |
| QA | Calidad / Testing | Validar que los CIs funcionen correctamente antes de producción |
| PROD | Producción | Ejecutar CIs en ambiente estable y controlado |

## Explicación de configuración

La base de datos se levantará en un contenedor de Docker mediante las siguientes variables de entorno.

|  |
| --- |
| DB\_HOST  DB\_USER  DB\_PASS |

## Inicialización DB

Con el entorno virtual de la DB de docker configurada es necesario levantar el contenedor, los siguientes comandos son los recomendados:

Levantar el proyecto en segundo plano y hacer el build.

* docker compose up -d --build

Eliminar los contenedores y volumenes

* docker compose down -v --rmi all

## Depliegue de entorno

Pasos para correr son:

|  |
| --- |
| * python -m venv venv * venv\Scripts\activate * pip install -r requirements.txt |

Esto ejecutara este api:

* <http://127.0.0.1:5000/>