🏁 Proyecto Inicial - Middleware ETL 🏁

En este README encontrarás los pasos iniciales para comenzar a trabajar con el proyecto principal y utilizarlo de manera correcta en tu entorno de trabajo; pero como bien dice el nombre, este proyecto es una fase inicial. Esto quiere dar a entender que el proyecto puede cambiar la metodología en como se descargaría o de por sí, un cambio total de la estructura de esta.

Conociendo nuestro proyecto 🧐

Para comenzar a trabajar con el proyecto necesitamos saber previamente las herramientas que utilizaremos, como también tener nocion de que versión estamos usando. Aquí una muestra de las dependencias mas importantes que usaremos:

```
Python: 3.12.4
Apache Airflow: 2.10.0
Great Epectations: 0.18.19
pandas: 2.1.4
Pyenv: 2.4.10
SqlAlchemy: 1.4.52
```

> Estas dependencias son especificamente las que Airflow recomienda usar con python 3.12 y Apache-Airflow 2.10.0

Ahora se mostrará la estructura de nuestro proyecto para un mejor entendimiento de la guia de instalación.

```
.
└── AIRFLOW/
    ├── workflow_management/
    │   ├── gx/
    │   │   ├── checkpoints/
    │   │   ├── expectations/
    │   │   │   └── my_expectation.json
    │   │   ├── plugins/
    │   │   ├── profilers/
    │   │   ├── uncommitted/
    │   │   └── great_expectations.yml
    │   └── dags/
    │       └── dag_cosume_validate.py
    ├── src/
    │   ├── modules/
    │   │   ├── extract/
    │   │   │   ├── data_extractor.py
    │   │   ├── transform/
    │   │   │   └── data_transformation.py
    │   │   └── load/
    │   │       └── data_loader.py
    │   └── common/
    │       ├── models/
    │       │   └── model.py
    │       ├── script_sql/
    │       └── utilities/
    │             └── data_cleaner.py
    │
```

```
    ├── logs/
    │   ├── dag_id=*
    │   ├── dag_procesor_manager
    │   └── scheduler
    ├── .env
    ├── .gitignore
    ├── .python-version
    ├── airflow.cfg
    ├── config.sh
    └── requirements.txt
```

Paso 1 - Instalación 🖇️

Comenzaremos descargando el proyecto directamente desde este repositorio, para eso, ejecutaremos el siguiente comando:

```
git clone https://github.com/Jhopcel/Middleware-ETL-APT.git
```

Una vez clonado el proyecto desde nuestro repositorio, ingresamos a la carpeta correspondiente del Proyecto, para esto navegaremos hacia la carpeta “middleware_capstone”.

Ya dentro, crearemos un ambiente virtual para descargar nuestras dependencias y hacer funcionar el proyecto, para esto ejecutamos el siguiente comando:

```
pyenv virtualenv 3.12.4 <nombre_ambiente>

#Ejemplo:

pyenv virtualenv 3.12.4 my_env
```

Ahora comenzaremos con la descarga de las dependencias del proyecto que se encuentra en requirements.txt, para esto ejecutamos el siguiente comando:

```
pip install -r requirements.txt --constraint
"https://raw.githubusercontent.com/apache/airflow/constraints-
2.10.0/constraints-3.12.txt"
```

Paso 2 - Configuración de nuestro entorno 🔨

Comenzamos configurando nuestras variables de entorno para la correcta ejecución de nuestros DAGs, para esto seguiremos la misma estructura que hay en ".env_example" que se encuentra en la carpeta, aquí te muestro de igual manera cuáles son las variables de entorno que utilizaremos:

```
CONNECTION_TO_DB="" #Esta es la conexión hacia tu base de datos, la sintaxis es
la siguiente: motorDB://userName:password@host/nameDatabase

LOCAL_CONNECTION_TO_METADATA_DB="" #Esta es la conexión por defecto y que se
aloja en la raiz del proyecto, la sintaxis default es:
sqlite:///home/user/my_proyect/airflow.db
```

Ahora vamos a necesitar indicarle Airflow donde está nuestro proyecto y sus configuraciones. Para esto ejecutaremos un archivo ".sh", que nos permitirá hacer esto de manera automática. Al ejecutar el archivo a demás de crear nuestras variables de entorno, también creará una variable con una "Cadena de conexión" para la base de datos de Airflow, esta cadena es un paso importante, ya que dentro tendrá todos los metadatos que Airflow necesita para funcionar. Esta

"cadena de conexión" se configura en el archivo `.env`, específicamente la variable "LOCAL_CONNECTION_TO_METADATA_DB". Si no cuentas con una conexión a la base de datos, solo deja la variable en blanco, y esto te conectará a una base de datos local, gracias al archivo "config.sh"

Para ejecutar el archivo solo ingresamos el siguiente comando:

```
source config.sh
```

> Por defecto Airflow asigna la carpeta `/airflow` como predeterminado, pero si tu carpeta raíz tiene un nombre diferente y no ejecutaste el archivo `config.sh`; cuando ejecutes algun comando exclusivo de airflow, este automaticamente creará la carpeta /airflow y se descargará todo en esa direccion, pero no en la tuya.

Configuraciónes BBDD

Ahora instalaremos las dependencias de nuestra base de datos de Airflow y la conexión mediante SqlAchemy. Para esto tendremos dos opciones de motores de base de datos:

* **Opción 1**: **MySQL**

```
sudo apt-get install -y build-essential libmysqlclient-dev

sudo apt-get install mysql-server

pip install mysql-connector-python==9.0.0

pip install mysqlclient==2.2.4


* **Opción 2**: **PostgreSQL (Por defecto y con la que esta construida el proyecto")**
```bash
sudo apt install postgresql postgresql-contrib
```

* Ingresamos a la consola con el usuario por defecto, que es "postgres":

```
sudo -u postgres psql
```

* Cambiamos la contraseña por defecto e ingresamos una nuestra:

```
\password
```

* Creamos la base de datos:

```
CREATE DATABASE test_middleware;
```

> Recuerda que este es un ejemplo, tú puedes cambiar el nombre de la base de datos a tu conveniencia

* ¡Listo!, ahora salimos de la consola con:

```
\q
```

Paso 3 - Iniciando nuestro DAG 🟢

Luego de realizar los pasos anteriores, comenzaremos iniciando el servidor de Airflow para poder tener una mejor percepción de nuestras tareas que se ejecutarán. Para iniciar el servidor de
```

manera correcta se necesitan 2 comandos que se deben ejecutar, para esto nos podemos apoyar de "visual studio code", ya que ofrece varias instancias de consolas. Aquí te muestro los pasos correspondientes para iniciar el servidor dentro de "visual studio code":

En nuestro WSL2 y dentro de la carpeta raiz del proyecto, ejecutaremos un comando que abrirá "visual studio code" en nuestro sistema local:

```
code .
```

Ya con el "Visual Estudio Code" abierto, nos posicionaremos en la parte superior, especificamente en la opción de "terminal" y seleccionamos la opción de "Nuevo Terminal".

> El paso anterior nos abrirá una terminal con la ruta de la carpeta en nuestro entorno local.

Ahora necesitamos nuestra instancia en WSL dentro de VSC, para esto nos vamos al símbolo de + y seleccionamos nuestra instancia actual.

Una vez hecho estos pasos podemos crear tantas instancias queramos. Pero ¡OJO! que estas instancias son diferentes a la sesión actual de nuestro WSL, ya que no mantiene las variables de entorno que creamos en la instancia principal, si queremos mantener la misma variable de entorno que asignamos anteriormente, solo debemos ejecutar el archivo `config.sh`

Ya que tenemos listo y configurado nuestro entorno, comenzaremos a ejecutar los comandos para iniciar el servidor. Estos comandos son los siguientes:

```
airflow webserver --port 8080

airflow scheduler
```

> Recurda que estos comandos se deben ejecutar en diferentes terminales.

Por ultimo, ingresaremos al servidor mediante cualquier navegador de nuestro sistema local, esto escribiendo el siguiente url:

```
http://localhost:8080/
```

¡Listo!, ya estarias dentro del servidor web y observando los DAGs que se encuentran en nuestro proyecto.