Guía de Configuración de Alembic con SQLAlchemy y PostgreSQL

Ordoñez Silva, Yonel Jr.

2025-11-19

Indice

I. Instalación de dependencias

```
pip install sqlalchemy psycopg2-binary alembic python-dotenv
```

- `sqlalchemy`: ORM para Python.
- `psycopg2-binary`: Driver PostgreSQL.
- `alembic`: Migraciones de base de datos.
- `python-dotenv`: Carga variables de entorno desde `.env`.

II. Creación de la estructura de Alembic

```
alembic init alembic
```

Esto crea:

```
alembic/
    env.py
    script.py.mako
    versions/
alembic.ini
```

Donde:

- `versions/`: Contendrá las migraciones.
- `env.py`: Configuración de conexión y carga de modelos.

III. Configuración de variables de entorno

Crea un archivo `.env` en la raíz del proyecto:

```
DATABASE_URL=postgresql://usuario:password@localhost:5432/mi_db
```

No guardes contraseñas en alembic.ini.

IV. Configuración de Alembic para usar .env

En `alembic/env.py`:

```
import os # Añadir os
from dotenv import load_dotenv # Añadir dotenv
from logging.config import fileConfig
from sqlalchemy import engine_from_config
from sqlalchemy import pool
from alembic import context

# Cargar variables del .env
load_dotenv()

# Alembic config
config = context.config
fileConfig(config.config_file_name)

# Asignar la URL desde .env
DATABASE_URL = os.getenv("DATABASE_URL")
```

```
config.set_main_option("sqlalchemy.url", DATABASE_URL)

# Importar modelos
from src.data_access_layer.models import Base  # Ajusta al path de tus modelos
# Aqui importa todos tus modelos
# por ejemplo:
# from src.data_access_layer.base import Base
# from src.data_access_layer.models.CargoModel import CargoModel
# from src.data_access_layer.models.CategoriaModel import CategoriaModel
# from src.data_access_layer.models.ComprobanteCargoModel import comprobante_cargo
# ...
target_metadata = Base.metadata

# Conexión y migraciones
def run_migrations_online():
    connectable = engine_from_config(
        config.get_section(config.config_ini_section),
        prefix='sqlalchemy.',
        poolclass=pool.NullPool
    )
    with connectable.connect() as connection:
        context.configure(
            connection=connection,
            target_metadata=target_metadata,
        )
        with context.begin_transaction():
            context.run_migrations()

if context.is_offline_mode():
    context.run_migrations()
else:
    run_migrations_online()
```

En `alembic.ini` deja la URL vacía:

```
sqlalchemy.url =
```

V. Configuración de `.gitignore`

```
# Entorno virtual
venv/
*.pyc

# Base de datos local
```

```
*.db
*.sqlite3

# Archivos de logs
*.log

# NO ignorar alembic ni versiones
# alembic/
```

Nota: Los archivos en `alembic/versions/` deben versionarse, o sea, no deben colocarse en el `.girignore`

VI. Flujo de trabajo para crear y aplicar migraciones

1. Crear migración después de cambiar modelos:

```
alembic revision --autogenerate -m "nombre_descriptivo"
```

Recomendaciones:

- Usa nombres descriptivos: add_producto_table, update_usuario_model, remove_old_column.
- Esto genera un archivo en `alembic/versions/`.

2. Aplicar migración a la base de datos:

```
alembic upgrade head
```

3. Revertir migración si es necesario:

```
alembic downgrade -1
```

- -1 → revierte una migración.
- También puedes usar el id de la revisión.

4. Ver historial de migraciones:

```
alembic history
```

VII. Buenas prácticas

- No poner credenciales en el repositorio, usar `.env`.
- Versionar todos los archivos de migración.
- Nombres descriptivos para revisiones.
- Probar migraciones en entorno de desarrollo antes de aplicar a producción.
- Mantener sincronizados los modelos y la base de datos.
- Revisar el contenido de autogenerate antes de aplicar migración.

VIII. Resumen del flujo

- Crear y configurar .env.
- Configurar `env.py` para usar `DATABASE_URL`.
- Crear modelos con `SQLAlchemy`.
- Crear migración con: `alembic revision --autogenerate -m "desc"`.
- Revisar el archivo generado en `alembic/versions/`.
- Aplicar migración con: `alembic upgrade head`.
- Mantener migraciones versionadas en Git y usar `.env` para credenciales.