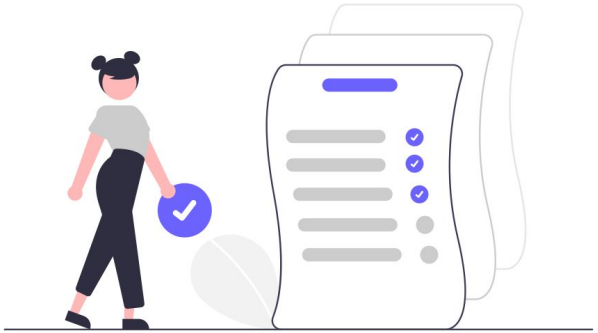
Express - PostgreSQL

DEV.F:
DESARROLLAMOS(PERSONAS);

¿Qué aprenderemos?



- Conectar expressJS a DB.
- Archivos .sql
- Uso de ORMs.

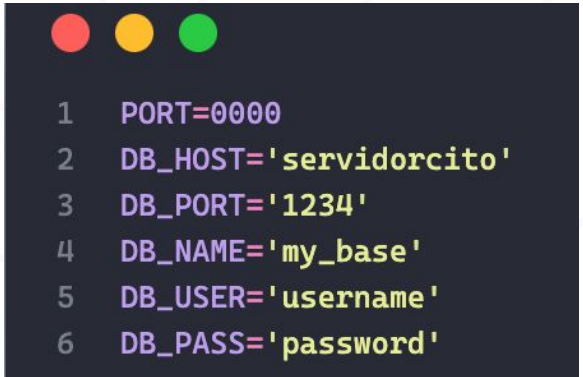
DEV.F::

DEV.F:

Conexión de Express a PostgreSQL

Consideraciones previas

1. Instalar la dependencia **dotenv**.
2. Creación de los archivos:
   a. **.env**: Contiene las variables de entorno del servidor. Este archivo no se sube al repo, si se sube al servidor.
   b. **.env.example**: Es una estructura de ejemplo del .env. . Este archivo si se sube al repo.
   c. **config**: Archivo dedicado a leer el .env mediante dotenv y exportar las claves leídas en el servidor.

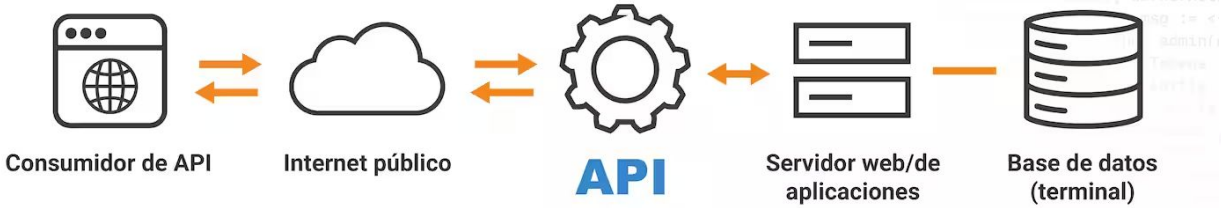


Consideraciones previas

3. Si la API va a recibir body de tipo json, cerciorarse de que se esté usando el middleware express.json() o body-parse.

4. Si la API se va a consumir desde otro servidor, configurar el acceso CORS.

5. Leer sobre las buenas prácticas de conexión a DB con express y postgres.



Conexión de Express a PostgreSQL

1. Contar con una BD en PostgreSQL.
2. Instalar la dependencia pg.
3. Crear un archivo de conexión, preferentemente de tipo pool.
4. Apuntar el consumo de datos al API a la DB.
    a. Previamente debe tener las claves de la DB en el .env
5. Utilizar sentencias query string de acuerdo a la sintaxis de pg para generar sus queries.

DEV.F

Otras alternativas

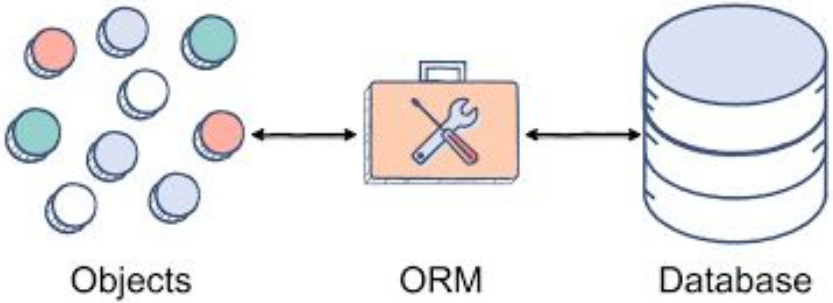
Si bien utilizar query strings es un estándar y es fácil para quien ha utilizado SQL toda su vida, suele presentar algunas complicaciones:

- Escalabilidad.
- Complejidad.
- Curva de aprendizaje.
- Alta cohesión al DBMS.

ORMs

Object Relational Mapping (ORMs)

Object Relational Mapping es una técnica de programación que nos permite trabajar con bases de datos relacionales también conocidos como RDMS (Relational Database Management System), como pueden ser SQL Server, Postgresql, MySQL, Oracle..., mediante código de POO escrito en nuestro lenguaje de programación.



Ventajas de los ORMs

- Menor curva de aprendizaje.
- Mejor escalabilidad y facilidad de migración.
- Más rápido en acceso a datos.
- Seguridad.
- Migrations y Seeders.
- Mejora la calidad y legibilidad del código.

ORMs y ODMs

- Un **orm** se relaciona con un modelo de objetos y una base de datos relacional.
- Un **odm** se relaciona con un modelo de objetos y una base de datos de documentos.

Doctrine

Docs

Docs

ExpressJS con Sequelize.

ExpressJS con Knex.

CRUD con Moongose.

¿Por qué usar un ORM? - Ventajas y desventajas