Automatización de reportes

API para automatización de generación de reportes de **Analítica.** El sistema es una API de Python y Flask que dispara la ejecución de los reportes cuando recibe una HTTP request, la cual es lanzada por un trigger de DB mediante un curl.

Una vez accionadas las funciones de ejecución de reportes, estos comenzarán a ser generados según las queries y configuraciones ubicadas en `reportes.json`, las cuales definen que parámetros usan, cuándo y con qué fechas son ejecutados, entre otras configuraciones.

Instalación

⚠️ **Importante estar localizado en el directorio principal del proyecto.**

Para instalar el script simplemente hay que ejecutar en la terminal el script de `instalar.bat`

```
./instalar.bat
```

Luego de eso se creará un entorno virtual de python y se instalarán las librerías necesarias para el funcionamiento del código. Finalmente mostrará un mensaje de confirmación.

Ejecución

⚠️ **Importante estar localizado en el directorio principal del proyecto.**

Para ejecutar la API necesitamos ejecutar el archivo de `ejecutar.bat` una vez ya se hayan instalado las librerías.

```
./ejecutar.bat
```

👉 **Nota: al ejecutar este programa, dado que es una API, no veremos una ejecución directa de los reportes y queries a la base de datos, más bien se verá un mensaje de que nuestro programa de servidor esta en modo `listen` esperando por una HTTP request.**

¿Cómo Agregar un reporte?

Los reportes se agregan en el archivo de `reportes.json`, que es la lista que se encarga de decirle al sistema **que reportes ejecutar? cuándo ejecutarlos y más parámetros personalizables.** Dentro de este archivo encontramos la siguiente estructura:

```
{
   "credenciales": {
      "usuario": "<usuario de sharepoint>",
      "contrasena": "<contraseña del usuario de sharepoint>"
   },

   "reportes_automaticos": {
      ...
   },


   "especiales": {
      ...
   }

}
```

- Las `credenciales` son las que se usan en los `reportes automáticos` para subirlos al sharepoint, los `especiales` tienen sus propias credenciales.
- Los `reportes_automaticos` y `especiales` contienen la lista de reportes así como sus respectivas configuraciones, su estructura está definida abajo así como guías para ayudarte a determinar donde colocar tu reporte nuevo.

¿Tu reporte es una simple query de SQL que usa fechas? `reporte automático`

```
"<nombre del reporte>": {
   "horario": "<puede ser: diario, semanal, mensual o trimestral>",
   "query": "<ruta al archivo de la query .sql>",
   "guardarcomo": "<nombre con el que guardar el archivo en sharepoint>.xlsx",
   "params": ["<variables del código a incrustar en la query, suelen ser fechas>"
   "sharepoint": {
      "site_url": "<url del sitio principal del sharepoint>",
      "relative_folder": "<ruta relativa al folder donde se va a guardar el reporte>"
   }
}
```

Sino, es un `Reporte especial` osea un script de Python en lugar de SQL

```
"especiales": {
      "<nombre del reporte>": {
         "horario": "<puede ser: diario, semanal, mensual o trimestral>",
         "script": "<ruta al archivo del script .py>",
         "params": "<en este caso los parametros se pasan como argumentos de
      }
   }
```

Funcionamiento

Detalles de conexiones: DB y Sharepoint

DB

Sharepoint

💾 Despliegue y Notas

- ⚠️ Actualmente dada que la conexión a la DB es mediante una `trusted connection` no se puede ejecutar el sistema en un docker, ***creo***, dado que la conexión es con el usuario de Microsoft del equipo.
- ⚙️ El desarrollo de este proyecto es en python, puramente un script ejecutable, aunque creo que podría ser una buena idea a futuro combinar el uso de `n8n` con este código, para tener mejores estándares y flujos de desarrollo y automatización.
- 💻 Creo que se debería de crear un usuario de DB específicamente para acceder como automatización, un rol que tenga únicamente permisos de lectura.