¿Qué patrón identificaron?

El patrón que identificamos fue el Singleton, un indicio a simple vista fue el nombre del archivo principal.

¿Dónde y cómo se aplica en el código?

Se aplica en la clase `DBConfigSingleton`
-En la instancia privada:

```
private static DBConfigSingleton instance;
```

-En el constructor privado:

```
private DBConfigSingleton()
```

-En el método de acceso estático:

```
public static synchronized DBConfigSingleton getInstance()
```

¿Qué problema resuelve este patrón en ese contexto?
El uso de este patrón de diseño resuelve dos problemas puntuales sobre nuestra problematica, relacionado con nuestra base de datos.
-**Garantiza una Instancia Única:** Asegura que **solo exista un objeto** de `DBConfigSingleton` en toda la aplicación. Esto es crucial para la configuración de la base de datos, ya que evita que diferentes partes del programa creen sus propias configuraciones, lo que podría llevar a inconsistencias o a un mal manejo de las conexiones.

-**Proporciona un Punto de Acceso Global:** Ofrece un **método global y controlado** (`getInstance()`) para acceder a esa única instancia. Cualquier parte del código que necesite abrir (`openConnection()`) o cerrar (`closeConnection()`) la conexión a la base de datos puede pedir esta instancia y estar segura de que está trabajando con el *mismo y único* gestor de conexión.