Análisis Lineal del Código: `facial_recognition_utils.py`

Este archivo, `facial_recognition_utils.py`, se dedica exclusivamente a las operaciones relacionadas con el reconocimiento facial. Sus dos responsabilidades principales son: 1) procesar imágenes de usuarios conocidos para crear "encodings" o representaciones numéricas de sus rostros, y 2) cargar estos encodings en la memoria para que el sistema pueda realizar comparaciones en tiempo real.

El script comienza importando las librerías necesarias. `import face_recognition` es la librería central que proporciona toda la funcionalidad para detectar y codificar rostros. `import pickle` se utiliza para guardar y cargar los datos de los encodings en un archivo binario, lo cual es mucho más eficiente que un archivo de texto. `import os` se usa para manejar rutas de archivos y directorios, y `import sqlite3` (o una función de `db_manager`) es necesario para interactuar con la base de datos y asociar los rostros con la información de los usuarios.

Se define una variable global, `encodings_faciales_cargados_global = []`, que es una lista que actuará como la memoria caché del sistema. Contendrá la información de todos los usuarios cuyos rostros han sido procesados y están listos para ser comparados.

La primera función clave es `crear_encodings_de_rostros_conocidos()`. Esta es una función de "entrenamiento" o preparación que se ejecuta típicamente una sola vez o cuando se añaden nuevos usuarios. Su trabajo es recorrer el diccionario `USUARIOS_DE_PRUEBA_IMAGENES`, que mapea nombres de usuarios a sus archivos de imagen (ej. "Fabrizio Reyes" -> "rostro_fabrizio.jpg"). Para cada usuario, carga la imagen desde la carpeta definida en `constants.ROSTROS_CONOCIDOS_DIR` usando `face_recognition.load_image_file()`. Luego, utiliza `face_recognition.face_encodings()` para analizar la imagen, encontrar el rostro y convertirlo en un `encoding`, que es un vector de 128 números que representa las características únicas de ese rostro. Si se encuentra un rostro, el `encoding` junto con el nombre del usuario se guarda en una lista. Una vez que se han procesado todas las imágenes, esta lista completa se guarda en un archivo `pickle` (definido por `constants.ARCHIVO_ENCODINGS_FACIALES_PKL`) usando la función `pickle.dump()`. Este archivo `pkl` es el resultado final del proceso de entrenamiento.

La segunda función esencial es `cargar_encodings_faciales_al_inicio()`. Esta función se llama al iniciar la máquina de estados. Su objetivo es leer el archivo `pkl` creado por la función anterior y cargar los datos en la variable global `encodings_faciales_cargados_global`. El proceso es el siguiente: abre el archivo `pkl` con `pickle.load()` para recuperar la lista de nombres y encodings. Luego, para cada nombre en la lista, realiza una consulta a la base de datos usando la función `obtener_usuario_por_nombre_bd()` para obtener toda la información de ese usuario (DNI, nivel, horario, etc.). Finalmente, fusiona la información de la base de datos con el `encoding` del rostro y lo añade a la lista en memoria. De esta forma, la variable global no solo contiene el `encoding`, sino todos los datos necesarios para tomar una decisión de acceso. Si un usuario del archivo `pkl` no se encuentra en la base de datos, se añade a la lista con datos limitados y se muestra una advertencia, permitiendo que el reconocimiento funcione, aunque sin reglas de negocio complejas.

El bloque final, `if __name__ == '__main__':`, permite que este script sea ejecutable por sí mismo. Esto es extremadamente útil para pruebas. Si ejecutas `python facial_recognition_utils.py`, el script intentará (re)generar el archivo de encodings llamando a `crear_encodings_de_rostros_conocidos()` y luego probará la carga de los mismos

con `cargar_encodings_faciales_al_inicio()`, imprimiendo los resultados en la consola. Esto permite verificar que el proceso de creación y carga de encodings funciona correctamente sin tener que lanzar toda la aplicación.