Proyecto CUIA:
Taylor's Experience

UNIVERSIDAD DE GRANADA

María de las Angustias Izquierdo García
Computación Ubicua e Inteligencia Ambiental

1. Documentación

Taylor’s Experiencie trata, tal y como se comentó en la propuesta de proyecto, de una aplicación en la que en función del estado de ánimo del usuario, recomienda una canción y la reproduce en Spotify, y además tiene la opción de probar el maquillaje temático dependiendo del disco que solicite el usuario. Por otra parte, el usuario también tiene la opción de añadir a favoritos canciones y también de añadir a no favoritos, para que en el caso de que se le recomiende una canción que se encuentra en no favoritos, se recomiende una nueva canción.

Registro de usuario e inicio de sesión

En primer lugar, al ejecutar la aplicación, se abre la cámara y el programa detecta si el usuario está registrado o no. Si se detecta un usuario registrado, se inicia sesión con ese usuario. En caso de que no esté registrado, se solicita por la terminal que el usuario introduzca un nombre de usuario y una contraseña, y tras esto se inicia sesión automáticamente.

Spotify

Para poder reproducir música en Spotify desde una aplicación, es necesario crearse una API. Los pasos se detallan en el manual de usuario. Una vez creada la API, obtenemos el CLIENT_ID y el CLIENT_SECRET, el cual utilizaremos en el archivo .env y cargaremos en los archivos necesarios haciendo lo siguiente:

```python
from dotenv import load_dotenv
load_dotenv()
client_id = os.getenv("CLIENT_ID")
client_secret = os.getenv("CLIENT_SECRET")
redirect_uri = os.getenv("REDIRECT_URI")
```

Luego, para poder realizar solicitudes a la API es necesario generar un token usando el client_id y el client_secret, el cual se genera en la función get_token de gettracks.py. La función combina el client_id y el client_secret en una cadena separada por dos puntos, la codifica en formato Base64 y prepara el encabezado de autenticación requerido. Luego, envía una solicitud POST a la URL de token de Spotify con los encabezados de autenticación y un cuerpo de datos que especifica el tipo de concesión como "client_credentials". La respuesta de esta solicitud contiene un token de acceso en formato JSON, del cual se extrae y devuelve el valor del token. Este token es necesario para autenticar y realizar futuras solicitudes a la API de Spotify.

Por último get_auth_header(token) toma un token de acceso como argumento y devuelve un diccionario con el encabezado de autorización necesario para realizar solicitudes autenticadas a la API de Spotify.

Recomendación de canción y reproducción

En primer lugar, para detectar la emoción del usuario he utilizado la librería DeepFace. Esta librería es capaz de detectar hasta 7 emociones: enfado, miedo, neutral, tristeza, disgusto, felicidad y sorpresa. Para poder obtener una canción de Taylor en función de la emoción, realicé una clasificación a mano creando una playlist para cada emoción en Spotify. Luego, para poder obtener la información de las canciones de cada playlist, me creé una API de Spotify, y luego realicé llamadas a la misma para obtener la información de cada playlist en un json diferente. Por lo tanto, tras la ejecución del archivo gettracks.py, obtuve 7 json diferentes en función de las emociones: taylor_angry_tracks.json, taylor_disgusted_tracks.json, taylor_happy_tracks.json, taylor_neutral_tracks.json, taylor_sad_tracks.json, taylor_scared_tracks.json y taylor_surprised_tracks.json. También obtuve el archivo all_taylor_songs.json el cual contiene todas las canciones de la artista, ya que la aplicación también permite reproducir la canción que el usuario desee sin la necesidad de que se detecte una emoción, por lo que para este caso las canciones se buscan en este archivo.

Para la detección de las emociones, en el archivo music_recommendation.py capturo el video desde la cámara web, analizo las emociones en diferentes frames utilizando hilos y DeepFace, y finalmente determino la emoción más frecuente en el video capturado.

Luego, una vez se detecta la emoción, se pasa a la función music_recommendation, la cual busca en el json correspondiente de la emoción una canción aleatoria y devuelve la URL y el nombre de la misma. Finalmente en el main, para reproducir la canción, se llama a la función play_track, la cual se encarga de obtener el id del dispositivo de la cuenta vinculada a la API, y luego utiliza la función start_playback de la API de Spotify.

Opciones del usuario

El usuario tiene las opciones de añadir a favoritos una canción que le haya gustado, y añadir a no favoritos las que no le hayan gustado, y en el caso de que se le recomiende una canción y esta pertenezca a no favoritos, se le recomendará una nueva.

Por otra parte, también tiene la opción de reproducir canciones desde favoritos y solicitar por comandos de voz cualquier canción que desee.

Programa principal

Para manejar la concurrencia entre el reconocimiento de voz y el resto de tareas he utilizado hilos. El hilo principal comienza con el registro del usuario mediante reconocimiento facial y luego crea un hilo separado para manejar continuamente los comandos de voz del usuario, lo que permite que la aplicación escuche comandos mientras realiza otras tareas. Este hilo de entrada del usuario se encarga de recibir comandos de voz y ponerlos en una cola para ser procesados. En el "user_dashboard", los

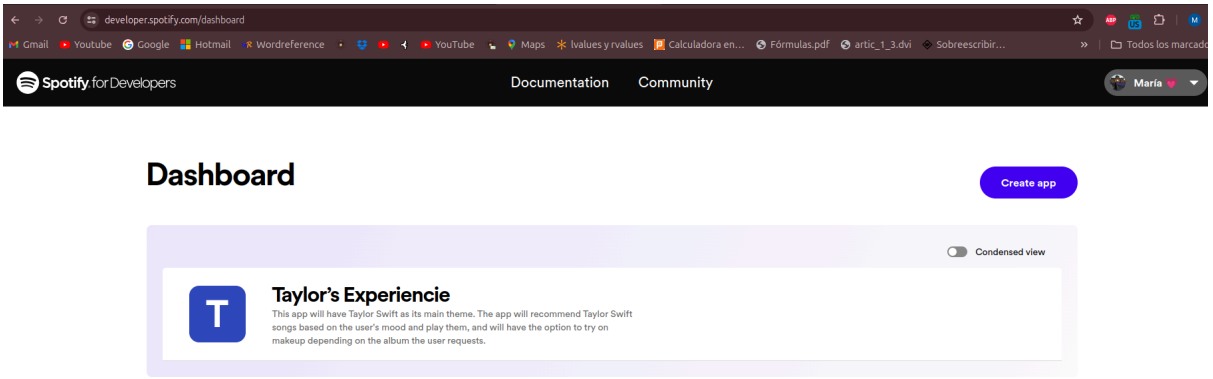
hilos permiten que el programa maneje la reproducción de música y la aplicación de maquillaje sin bloquear otras operaciones. Cuando se elige reproducir música, se puede iniciar un nuevo hilo para reproducir la música basada en la emoción detectada, mientras que otro hilo puede seguir escuchando comandos de voz para detener la música, cambiar de canción, añadir a favoritos, entre otros. De manera similar, cuando se selecciona la opción de aplicar maquillaje, el programa usa la cámara para capturar frames y aplicar efectos de maquillaje en tiempo real, permitiendo cambiar entre diferentes tipos de maquillaje según los comandos recibidos.

2. Manual de usuario

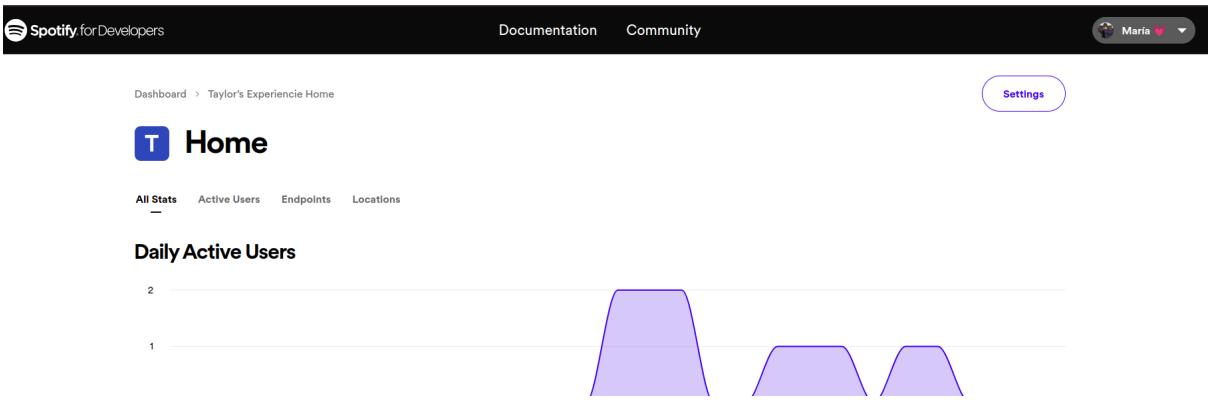
En primer lugar, para poder reproducir música en Spotify, hay que crearse una API accediendo al siguiente enlace:

https://developer.spotify.com/documentation/web-api/concepts/apps

en el mismo aparecen los pasos para crearse la API. En Redirect URI ponemos https://localhost:8000. Una vez creada, accedemos a https://developer.spotify.com/dashboard :



Le damos a nuestra API y luego a Settings:



En settings encontraremos nuestro CLIENT_ID y CLIENT_SECRET. Estos dos valores son los que tendremos que poner en el archivo .env del proyecto para que la música se reproduzca en nuestra cuenta de Spotify. En REDIRECT_URI del archivo .env pondremos https://localhost:8000.

Funcionamiento

Esta aplicación ha sido implementada con reconocimiento de voz, por lo que la mayoría de interacciones con el usuario es a través de reconocimiento de voz.

El funcionamiento de la aplicación es de la siguiente forma:

1. Registro o inicio de sesión: Cuando se ejecuta la aplicación, se inicia el proceso de registro. La aplicación intentará reconocer tu rostro. Si no te reconoce, procederá a registrarte como un nuevo usuario.
2. Iniciar Menú: Después del registro, se mostrará un menú con tres opciones:

    1. Reproducir música

       Si el comando reconocido es reproducir música, se abre la cámara y te detecta una emoción y reproduce la canción en Spotify. Para que la reproducción se realice, es necesario tener abierta la aplicación de Spotify (o navegador) y haber reproducido una canción recientemente. Una vez se está reproduciendo, el usuario puede parar la canción pulsando Enter, y tras eso se inicia el reconocimiento de voz, teniendo el usuario las siguientes opciones:

        a. Nueva canción → se vuelve a reconocer la emocion del usuario y se reproduce una canción acorde a su emoción
        b. Añadir a favoritos → añade a favoritos del usuario la canción que se acaba de reproducir
        c. Añadir a no favoritos → añade a no favoritos del usuario la canción que se acaba de reproducir
        d. Reproducir de favoritos → reproduce una canción aleatoria de las favoritas del usuario
        e. Cerrar reproductor → nos lleva al menú principal en el que podremos volver a escoger: reproducir música, aplicar maquillaje o cerrar sesión
        f. Cerrar sesión → se cierra la aplicación
        g. Decir cualquier otra palabra, en cuyo caso el programa buscará si lo que se ha dicho coincide con una canción de Taylor, y si es el caso, se reproduce
    2. Aplicar maquillaje

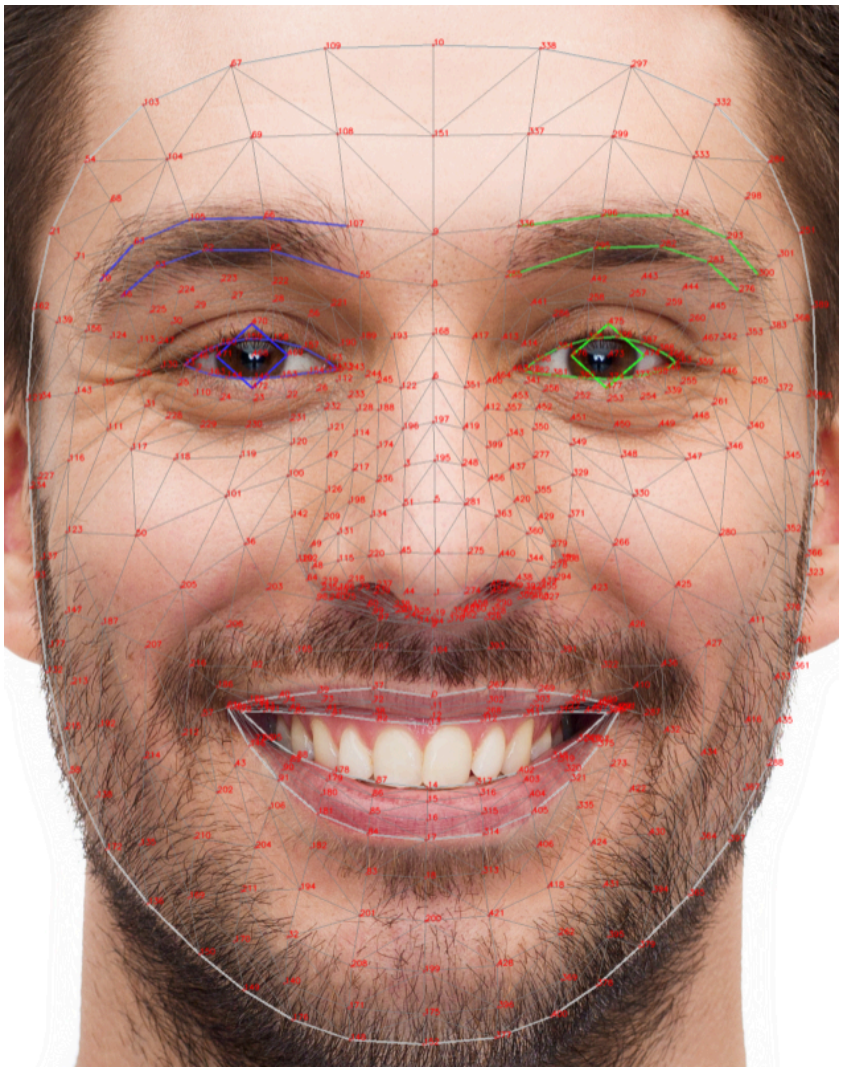
       Si el comando reconocido es aplicar maquillaje, se abre la cámara automáticamente y se espera que el usuario diga uno de los maquillajes disponibles: speak now, fearless, red makeup, lover o evermore. El reconocimiento de voz está siempre abierto para que el usuario pueda ir cambiando de maquillaje sólo con decir cual es el nuevo que quiere, y la cámara se mantiene abierta todo el tiempo.
       Si se reconoce cerrar sesión maquillaje, nos lleva al menú principal en el que podremos volver a escoger: reproducir música, aplicar maquillaje o cerrar sesión.

    3. Cerrar sesión: termina la ejecución de la aplicación

3. Dependencias

Para poder realizar con las funciones necesarias esta práctica he usado las siguientes librerías:

- mediapipe: esta es la biblioteca que he utilizado para poder implementar la funcionalidad del maquillaje. Permite obtener puntos de una cara a través de la siguiente plantilla:



- cv2: para procesamiento de imágenes y vídeos
- spotipy: para interactuar con la API de Spotify.
    - spotipy.Spotify: El cliente principal de Spotipy para hacer llamadas a la API de Spotify.
    - spotipy.oauth2.SpotifyOAuth: Clase para manejar la autenticación OAuth con Spotify.
    - spotipy.exceptions.SpotifyException: Clase para manejar excepciones específicas de Spotify.
- os: para leer variables de entorno
    - os.getenv: Función para obtener el valor de una variable de entorno.
- json: para trabajar con JSON

    - json.load: Función para leer y parsear archivos JSON.
- random
- dotenv: para cargar variables de entorno desde un archivo .env.
    - dotenv.load_dotenv: Función para cargar variables de entorno desde un archivo .env.
- speech_recognition:
    - speech_recognition.Recognizer: para el reconocimiento de voz.
    - speech_recognition.Microphone: para utilizar el micrófono como fuente de audio.
- unidecode: para normalizar texto y eliminar acentos y otros caracteres especiales.
- threading: para crear y manejar hilos de ejecución.
- queue
- sys
- face_recognition: para detectar, identificar y comparar caras en imágenes.
- time: proporciona funciones para trabajar con tiempo, como obtener la hora actual, hacer pausas en la ejecución.
- numpy: proporciona soporte para arrays multidimensionales y matrices, junto con una colección de funciones matemáticas de alto nivel para operar con estos arrays.
- base64: para codificar y decodificar datos utilizando la codificación Base64.
- requests: para envío de solicitudes HTTP/1.1 con métodos como GET y POST.
- math
- deepface
- collections