Para lanzar la aplicación debe tener instalado Docker.

Funcionamiento usuario :
Una vez descargado el código desde el repositorio en github, diríjase a la carpeta dockercode, una vez dentro de la carpeta abra una terminal y corra el mando *docker-compose up.* Espere 1 minuto mientras se inician los dos contenedores. Una vez pasado el minuto abra su navegador de preferencia y en una pestaña diríjase a la ruta http://localhost:5000/ presione login, ingrese a la cuenta de Google que desee y otorgue permisos a la aplicación. La aplicación reconocerá los archivos públicos asociados a su cuenta y los cambiará a privados. Una vez terminado el login y habiendo otorgado permisos a la aplicación usted será redirigido a la ruta http://localhost:5000/readfiles donde podrá observar todos los archivos asociados a su cuenta de Google, junto con los campos que pide este challenge. La primera vez usted encontrará archivos con accesibilidad Publica, pero una vez usted presione el botón Update files, estos se actualizarán de acuerdo al requerimiento de este challenge. Usted podrá hacer cambios a sus archivos de Google Drive y luego actualizar estos cambios presionando el botón Update files.

Funcionamiento técnico:
Una vez usted corre el comando *docker-compose up* se inicia primero una contenedor con una imagen MySQL que expone el puerto 3306 para las conexiones. Se crea una tabla llamada DriveFile que contiene los campos requeridos para este code challenge más el campo FileID que funciona como un campo único y sirve para poder diferenciar cada archivo (dado que en Google Drive este campo también es único). Una vez se construye el contenedor MySQL, se inicia un contenedor con una imagen ubuntu, donde se instala Python3,Flask y todas las librerías relacionadas con Google Drive API y Gmail API. Una vez se termina de construir el contenedor se lanza un servidor de Flask corriendo en localhost. Este contenedor de ubuntu expone el puerto 5000 para la conexión de regreso (callback, explicación más adelante) una vez se termina de realizar el flujo de login y otorgamiento de permisos a la aplicación.

Una vez se terminan de construir los dos contenedores, puede visitar su navegador de preferencia y visitar http://localhost:5000/ y presionar login. Login lanza el flujo para el proceso de login, para ello usa las credenciales de la aplicación almacenadas en el archivo client_secret.json, al terminal el flujo de login y otorgamiento de permisos, se necesita una url a la cual el flujo pueda volver (callback), por esa esta razón es que se expone un puerto (5000) junto con la url /callback (ver archivo redirect_uris en client_secret.json). Cuando se termina el flujo también se guarda un archivo llamado token.json necesario para poder llamar a las APIs. La ruta callback también llama la función fetch_start_page_token, necesaria para iniciar el token que lleva el tracking de los cambios de los archivos.

Desincronización y hilos:
Cuando el usuario configura un archivo como Público en su Drive y luego presiona el botón Update files, verá que el archivo aparecerá **aun** como Público en la ruta /readfiles lo cual no se alinea con los requisitos del ejercicio (puesto que pide que todo archivo público sea cambiado a privado), sin embargo, esto es necesario (de acuerdo a **mi** implementación) dado que de lo contrario el usuario siempre veria los archivos como privados y tendría que esperar de forma ***sincrónica*** hasta que se cambiaran uno a uno los permisos públicos a privados y se enviará uno a uno los correos a los respectivos dueños, lo cual podría

demorar cierto tiempo dependiendo de la cantidad de archivo publicos que este tenga, resultando en una mala experiencia de usuario, puesto que tendría que esperar todo este tiempo hasta que se le muestre la ruta /readfiles (ruta que muestra los archivos en la base de datos MySQL). Por tal razón, se ***evita todo este tiempo de espera*** usando hilos (threads) que de manera ***asincrónica*** van cambiando los permisos y enviando los correos a los respectivos dueños en un hilo aparte al de ejecución principal, por lo que el usuario verá los archivo aun como públicos hasta que vuelva a presionar el botón update files donde ya verá los cambios efectuados.