Tarea 3 y 4: Introducción al análisis y procesamiento de audio con Python.

Configurar el repositorio de GitHub, conda y JupyterLab.

En este apartado, se configurarán las herramientas para poder trabajar y realizar la tarea correctamente sin ningún tipo de problema, para ello, se seguirán los pasos del fichero 'README.md' del repositorio remoto https://github.com/mhaut/uex-audiopy.

```
(base) adrianlancho@adrianlancho-VirtualBox:~/Documentos$ git clone https://github.com/adlanchod/Repositorio1
Clonando en 'Repositorio1'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Recibiendo objetos: 100% (3/3), listo.
```

Imagen 1: Se puede ver como hemos creado un repositorio vacío llamado 'Repositorio1' y lo estamos clonando a nuestro PC para trabajar con el.

```
(mi_entorno) adrianlancho@adrianlancho-VirtualBox:~/Documentos$ jupyter lab
[I 2025-03-04 17:44:54.832 ServerApp] jupyter_lsp | extension was successfully linked.
[I 2025-03-04 17:44:54.840 ServerApp] jupyter_server_terminals | extension was successfully linked.
[I 2025-03-04 17:44:54.847 ServerApp] jupyterlab | extension was successfully linked.
[I 2025-03-04 17:44:54.850 ServerApp] Writing Jupyter server cookie secret to /home/adrianlancho/.local/share/jupyter/runtime/jupyter_cookie_secret
[I 2025-03-04 17:44:55.123 ServerApp] notebook_shim | extension was successfully linked.
[I 2025-03-04 17:44:55.172 ServerApp] notebook_shim | extension was successfully loaded.
[I 2025-03-04 17:44:55.183 ServerApp] jupyter_lsp | extension was successfully loaded.
[I 2025-03-04 17:44:55.188 ServerApp] jupyter_server_terminals | extension was successfully loaded.
[I 2025-03-04 17:44:55.191 LabApp] JupyterLab extension loaded from /home/adrianlancho/anaconda3/envs/mi_entorno/lib/python3.10/site-packages/jupyterlab
[I 2025-03-04 17:44:55.192 LabApp] JupyterLab application directory is /home/adrianlancho/anaconda3/envs/mi_entorno/share/jupyter/lab
[I 2025-03-04 17:44:55.193 LabApp] Extension Manager is 'pypi'.
[I 2025-03-04 17:44:55.268 ServerApp] jupyterlab | extension was successfully loaded.
[I 2025-03-04 17:44:55.271 ServerApp] Serving notebooks from local directory: /home/adrianlancho/Documentos
[I 2025-03-04 17:44:55.271 ServerApp] Jupyter Server 2.15.0 is running at:
[I 2025-03-04 17:44:55.271 ServerApp] http://localhost:8888/lab?token=c5625643e470a94f11df2d67885465926cac09ebc7057a6c
[I 2025-03-04 17:44:55.271 ServerApp]     http://127.0.0.1:8888/lab?token=c5625643e470a94f11df2d67885465926cac09ebc7057a6c
[I 2025-03-04 17:44:55.271 ServerApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
[C 2025-03-04 17:44:55.517 ServerApp]

    To access the server, open this file in a browser:
        file:///home/adrianlancho/.local/share/jupyter/runtime/jpserver-19500-open.html
    Or copy and paste one of these URLs:
        http://localhost:8888/lab?token=c5625643e470a94f11df2d67885465926cac09ebc7057a6c
        http://127.0.0.1:8888/lab?token=c5625643e470a94f11df2d67885465926cac09ebc7057a6c
[I 2025-03-04 17:44:55.557 ServerApp] Skipped non-installed server(s): bash-language-server, dockerfile-language-server-nodejs, javascript-typescript-langserver, jedi
anguageserver, sql-language-server, texlab, typescript-language-server, unified-language-server, vscode-css-languageserver-bin, vscode-html-languageserver-bin, vscode
[W 2025-03-04 17:45:02.658 LabApp] Could not determine jupyterlab build status without nodejs
```

Imagen 2: En esta imagen, se puede ver como iniciamos la herramienta Jupyter-lab para realizar las tareas propuestas.

Es importante mencionar que, para poder tener todas las dependencias y paquetes necesarios para la tarea crearemos un entorno de conda a partir del fichero 'environment.yml'. El entorno que se ha creado para realizar las tareas tiene el nombre de 'prueba'.

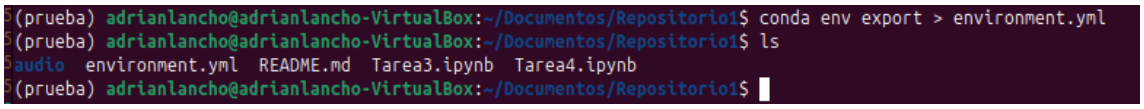
Análisis de audio con Python y Jupyterlab:

En este apartado, cada uno de los pasos, se encuentran explicados en el notebook 'Tarea3.ipynb' en el siguiente repositorio https://github.com/adlanchod/Repositorio1/tree/main.

Comprobar que no haya fallos:

A continuación, se realizarán los siguientes pasos:

- Kernel → Restart Kernel and Run All Cells para reiniciar el kernel y comprobar que no haya fallos.
- Kernel → Restart Kernel and Clear All Outputs. Esto se realiza para reducir el tamaño del fichero y poder sincronizarlo sin problemas.
- Exportaremos los paquetes instalados en nuestro entorno de conda creando el fichero 'environment.yml', utilizando el comando 'conda env export > environment.yml'(Imagen3).
- Y por último, sincronizaremos todo con GitHub.



Procesamiento de audio con Python y JupyterLab:

En este apartado, cada uno de los pasos, se encuentran explicados en el notebook 'Tarea4.ipynb' en el siguiente repositorio https://github.com/adlanchod/Repositorio1/tree/main.

URL del repositorio implementado:

https://github.com/adlanchod/Repositorio1/tree/main

1.Explicar con tus palabras: diferencia entre audio estéreo y mono.

La diferencia es que el audio mono, solamente se escucha por un canal, es decir, si tenemos unos auriculares se va a escuchar exactamente igual por el lado derecho que por el lado izquierdo, mientras que, el audio estéreo se escucha por dos canales, es decir, se pueden escuchar sonidos diferentes en el auricular derecho que en el izquierdo.

2.Investigar sobre la frecuencia de muestreo.

La frecuencia de muestreo es la cantidad de muestras que se producen por segundo en una señal analógica para convertirla en una señal digital. Se mide en hercios (Hz) o muestras oír segundo.

Funciona de la siguiente manera, cuando una señal analógica (como el sonido) es capturada, por ejemplo, por un micrófono, esta es continua en el tiempo. Para almacenarla o procesarla digitalmente, se necesita convertirla en valores discretos mediante un proceso llamado muestreo.

Un aspecto importante, es que para evitar pérdida de información y distorsión en la reconstrucción de la señal, la frecuencia de muestreo debe ser al menos el doble de la frecuencia máxima presente en la señal original, esto evita el aliasing.

3.Explicar: frecuencia de muestreo, aliasing, profundidad de bits, ancho de banda y tasa de bits.

Frecuencia de muestreo: Se trata del número de veces que se captura una muestra de una señal de audio.

Aliasing: Se trata de un error que se produce cuando la frecuencia de muestreo es demasiado baja. Como consecuencia, se pueden generar sonidos falsos o distorsionados.

Profundidad de bits: Se refiere al número de bits usados para representar cada muestra de audio.

Ancho de Banda: Es el rango de frecuencias que un sistema puede capturar o reproducir.

Tasa de Bits (Bitrate): Es la cantidad de datos procesador por segundo en un archivo de audio, cuanto mayor sea la tasa de bits, mayor es la calidad del audio.